Efficient Mode Selection Scheme for Video Coding

Md. Salah Uddin Yusuf

Department of Electrical and Electronic Engineering

Khulna University of Engineering and Technology

This dissertation is submitted for the degree of

Doctor of Philosophy

November 2016

I would like to dedicate this PhD thesis to my loving Parents,
my wife Tania Sultana and daughter Ariba Yusuf TUBA
with love. . . .

Declaration

This is to certify that the thesis work entitled **“Efficient Mode Selection Scheme for Video Coding"** has been carried out by **Md. Salah Uddin Yusuf** in the Department of Electrical and Electronic Engineering, Khulna University of Engineering and Technology, Khulna, Bangladesh. The above thesis work or any part of this work has not been submitted anywhere for the award of any degree or diploma.

Md. Salah Uddin Yusuf
November 2016

Acknowledgements

At first, I would like to prostrate in worship to Almighty Allah for helping me at every stages of my life, especially giving me the mercy, patience and ability to complete my PhD program successfully. Writing this part of thesis also gives me a formal opportunity to thank the people who have supported me and consequently had influence on the accomplishment of this research for last few years.

Foremost, I would like to express my deepest gratitude and respect to my honorable supervisor, Professor Dr. Mohiuddin Ahmad, for his invaluable guidance, excellent support, stimulating suggestions and encouragement, which helped me accomplish the doctorate degree and prepared me to achieve more life goals in the future. His extensive and up-to-date experience in the field of video processing has been invaluable in the process of exploring and evaluating new ideas. Special thanks are dedicated for the discussions we had on almost every working day during my research period and for reviewing my thesis.

Besides my supervisor, I would also like to thank my respected Doctoral Scrutiny Committee(DSC) members: Dr. Md. shahjahan, Head of the department of EEE, Professor Dr. M A Samad, Professor Dr. Md. Rafiqul Islam, Professor Dr. Abdur Rafiq and Professor Dr. Md. Al-Amin Bhuiya for their valuable comments and suggestions at different meetings, encouragement, insightful comments, and taking their valuable time in being part of thesis committee. I would like to thanks Professor Dr. Mohammad Shorif Uddin and Professor Dr. Mohammad Abdus Salam for being a part of my Board of Examiner of PhD. Special thanks are also due to Monira Islam, Assistant Professor, department of EEE, KUET for supporting and encouraging me always to accomplish this PhD research. Moreover, I would like to show my appreciation to my colleagues and all of my friends at KUET, who have supported and encouraged me over the past few years.

Last but not the least I am indebted to my family as all of them have patiently supported me throughout my entire academic journey. It is impossible to express my love and appreciation to my wife Tania Sultana and my little angel Ariba Yusuf Tuba in a word. I also would like to express my love and gratitude to my sister and brother-in-law. Finally, my deepest gratitude goes to my dear father and mother for providing their constant encouragement both financial and spiritual support throughout my life and who have devoted their everything to my education with invariant love and enthusiasm.

Certificate

This is to certify that the thesis titled **"Efficient Mode Selection Scheme for Video Coding"** by **Md. Salah Uddin Yusuf**, submitted to the Department of Electrical and Electronic Engineering, Khulna University of Engineering & Technology (KUET), Khulna, Bangladesh, for the award of Doctor of Philosophy (Ph.D.) carried out by him under my supervision. I certify that this thesis fulfils part of the requirements for the award of the degree. The thesis has not been submitted for any other degree or diploma elsewhere.

(Professor Dr. Mohiuddin Ahmad)
Department of Electrical and Electronic Engineering,
Khulna University of Engineering & Technology (KUET),
Khulna-9203, Bangladesh

Declaration of Publication

I hereby declare that this thesis incorporates the outcome of a research undertaken by me under the supervision of Professor Dr. Mohiuddin Ahmad. The contribution is covered in Chapter 3, 4, 5, 6, 7 and 8 of the thesis. In all cases, the key ideas, primary contributions, experimental designs, data analysis and interpretation, were performed by the author, and the contribution of co-author was primarily through the provision of valuable suggestions and helping in comprehensive analysis of the simulation results for publication. This thesis includes following papers that have been previously published/Accepted for publication/Submitted for publication in peer reviewed journals and conferences, as follows:

Thesis Chapter	Publication title/full citation	Publication status*
Chapter 3	Md. Salah Uddin Yusuf and Mohiuddin Ahmad, "Modified Intra 4×4 Prediction Mode Scheme For H.264/AVC Video Coding", Proceedings of the International Conference on Engineering Research, Innovation and Education (ICERIE), pp. 648-651, 2011.	Published
Chapter 4	Md. Salah Uddin Yusuf and Mohiuddin Ahmad, “A New Z-Scanning Scheme for Directional Spatial Prediction of AVS Intra Coding.” Proceedings of the International Conference on Computer and Information Technology (ICCIT), 2010.	Published
	Md. Salah Uddin Yusuf and Mohiuddin Ahmad, “A Novel Scanning Scheme for Directional Spatial Prediction of AVS Intra coding”, International Journal of Advanced Research in Electrical, Electronic and Instrumentation Engineering (IJAREEIE), vol.2, no. 8, pp. 4120-4126, August 2013	Published
Chapter 5	Md. Salah Uddin Yusuf and Mohiuddin Ahmad, “A Novel Rate-Based Macroblock Classification for Fast Mode Decision in H.264 Video Coding”, Proceedings of the International Conference on Computer and Information Technology (ICCIT), 2011.	Published
	Md. Salah Uddin Yusuf and Mohiuddin Ahmad, “A Novel Rate based Macroblock Classification for Fast Mode Decision in h.264 Video Coding”, International Journal of Advanced Research in Electrical, Electronic and Instrumentation Engineering (IJAREEIE), vol.2, no. 8, pp. 4111-4119, August 2013	Published

Chapter 6	Md. Salah Uddin Yusuf and Mohiuddin Ahmad, “Relative SAD-based Inter Mode Decision Algorithm for H.264/AVC Video Standard,” Proceedings of the International Conference on Computer and Information Technology (ICCIT), 2010.	Published
	Md. Salah Uddin Yusuf and Mohiuddin Ahmad, "Fast Inter Mode Decision Algorithm For H.264/AVC Video Standard, Proceedings of the International Conference on Engineering Research, Innovation and Education (ICERIE),pp. 590-595, 2011	Published
	Md. Salah Uddin Yusuf and Mohiuddin Ahmad, “Relative SAD-based Inter Mode Decision Algorithm for H.264/AVC Video Standard”, Journal of Multimedia, Academy Publishers, vol. 6, no. 5, pp. 425-435, 2011.	Published
Chapter 7	Md. Salah Uddin Yusuf, M. I. Hossain and Mohiuddin Ahmad, “Probability Based Fast Inter Mode Decision Algorithm for H.264/AVC Video Standard”, IFOST, 2014.	Published
	Md. Salah Uddin Yusuf, M. I. Hossain and Mohiuddin Ahmad, “A New Fast Inter Mode Decision Algorithm in H.264/AVC Video Coding”, 8th International Conference on Electrical and Computer Engineering (ICECE), pp. 278-281 December, 2014.	Published
	Md. Salah Uddin Yusuf, M. I. Hossain and Mohiuddin Ahmad, M. M. Rahman,N. Ahmed and M. A. Rashid, “Probability Based Fast Inter Mode Decision Algorithm for H.264/AVC Video Standard”, in International Journal of Applied Mathematics & Information Sciences, vol. 10, no.3, pp. 1071-1079, 2016.	Published
Chapter 8	Md. Salah Uddin Yusuf and Mohiuddin Ahmad, “ Efficient Search Point Reduction and Ranking of commonly used Motion Estimation Algorithms for Video coding”, Proceedings of the 2nd International Conference on Electrical Engineering and Information Communication Technology (ICEEICT 2015)	Published
	Md. Salah Uddin Yusuf, M. Islam, Mohiuddin Ahmad, "Fast and Efficient Inter Mode Decision Algorithm for H.264/AVC Encoding Optimization," in International Journal of Electronics Communication and Computer Engineering, vol. 6, no. 6, pp. 746-751, 2015	Published
	Md. Salah Uddin Yusuf, M. Islam, Mohiuddin Ahmad, "Efficient Search Points Reduction and Ranking of Motion Estimation Algorithms in Video Coding," in International Journal of Electronics Communication and Computer Engineering, vol. 6, no. 6, pp. 739-745, 2015.	Published

Abstract

Video coding plays vital role in communications and multimedia applications. The appearance and development of various new multimedia services have need for higher coding efficiency. H.264/AVC is the latest international video coding standard that can achieve considerably highest coding efficiency than previous standards and uses rate-distortion optimized (RDO) mode selection technique. This means that the encoder has to code the video by exhaustively trying all the mode combinations including the different intra and inter prediction modes. However, this mode selection process also makes the encoding process extremely complex, especially in the computation of the rate-distortion cost function, which includes the computations of the sum of squared difference (SSD) between the original and reconstructed image blocks and context-based entropy coding of the block. So, the highly complex mode selection process limits the practical implementation of codec's.

This dissertation proposes six different fast and efficient schemes to reduce the computational complexity of mode (Intra and Inter) selection process. It also can effectively reduce the spatial and temporal redundancy. In H.264/AVC intra coding, DC mode is used to predict regions with no unified direction and the predicted pixel values are same and thus smooth the varying regions which are not well de-correlated. A modified DC prediction (MDCP) mode is suggested to replace the existing DC mode based on the distance between the predicted and reference pixels. By analyzing the impact of scanning, a new Z-scan scheme is proposed which can remarkably reduce bit rates compared with AVS codec using conventional zigzag scan, while the PSNR of video sequences are maintained. The entire macroblock is divided into simple motion or complex motion contents and their cost functions are calculated by using the developed rate-based fast and efficient mode selection scheme, which plays a vital role as no additional operation module is required. To avoid the entropy coding method during intra and inter mode decision, a rate estimation method proposed and readjusted to fit the characteristic of low complexity distortion functions of SATD and SAITD that use more accurate estimation while achieving higher prediction performance with low computational requirement of feature. In addition, a relative sum of absolute difference (RSAD) based inter mode decision algorithm is proposed to determine the best inter mode without calculating rate-distortion costs. To further speedup SKIP mode early termination algorithm is adopted. Based on the probability characteristics, a fast inter mode decision algorithm is suggested which can determine the best inter mode only by calculating the Hadamard Transform (H-SAD) cost instead of rate distortion cost and avoid most of the complex computation processes like quantization, variable length coding, pixel reconstruction etc. To make the motion estimation process fast, a search point reduction scheme is introduced. Then the ranking and performance of the commonly used scheme are analyzed for different kinds of video. Experimental results indicate that the proposed efficient mode selection schemes reduce the computational complexity of H.264/AVC codec without sacrificing the quality of video which can be applied in implementing real time encoder for low-power mobile devices and general purpose computers.

Table of contents

List of figures

List of tables

List of abbreviations

3D	Three Dimensions
3SS	Three Step Search
4SS	Four Step Search
ANM	Adaptive Numbers of Modes
APDS	Adjustable Partial Distortion Search
ASR	Adaptive Search Range
AVC	Advanced Video Coding
BDM	Block Distortion Measure
BIP	Bi-directional Intra Prediction
BMA	Block-Matching Algorithm
BP	Baseline Profile
CABAC	Context-adaptive binary arithmetic coding
CAVLC	Context-Adaptive Variable Length Coding
CD-ROM . . .	Compact Disc Read Only Memory
CDS	Cross Diamond Search
CIF	Common Intermediate Format
DCT	Discrete Cosine Transform
DHS-NPDS . .	Dual halfway stop Normalized Partial Distortion Search
DMB	Digital Multimedia Broadcasting
DSL	Digital Subscriber Line
DS	Diamond Search
DVB-H	Digital Video Broadcasting-Hand held

DVD	Digital Versatile Disk
DWP	Distance based Weighted Prediction
EHEXBS	Enhanced Hexagon Based Search
EPDS	Edge based Partial Distortion Search
ESATD	Enhanced Sum of Absolute Integer Transform Differences
ET	Early Termination
FBMA	Fast Block-Matching Algorithm
FFSSG	Fast Full Search with Sorting by Gradient
FHT	Fast Hadamard Transform
FS	Full Search
HD	High-Definition
HEVC	High Efficiency Video Coding
HEXBS	Hexagon Based Search
HPDS	Hadamard transform based Partial Distortion Search
IBS	Intra mode Bits Skip
IDCP	Improved DC Prediction
IMBR	Intra Mode Bit Reduction
IPTV	Internet TV
ISO	International Organization for Standardization
ITU-T	International Telecommunications Union, Telecommunication Standardization Sector
JM	Joint Model (H.264/AVC)
JVT	Joint Video Team
MB	Macroblock
ME	Motion Estimation
MF	Multiplication Factor
MPEG	Moving Picture Expert Group
MPM	Most Probable Mode
MP	Main Profile

MV	Motion Vector
MVC	Multi-view Video Coding
MVD	Motion Vector difference
MVP	Motion Vector Predictor
NDS	New Diamond Search
NPDS	Normalized Partial Distortion Search
PDS	Partial Distortion Search
PSNR	Peak Signal to Noise Ratio
PSTN	Public Switched Telephone Network
QCIF	Quarter Common Intermediate Format
QP	Quantization Parameter
RD	Rate-Distortion
RDO	Rate-Distortion Optimization
SAITD	Sum of Absolute Integer Transform Differences
SD	Standard-Definition
SAD	Sum of Absolute Differences
SATD	Sum of Absolute Hadamard-Transform Differences
SSD	Sum of Squared Differences
SVC	Scalable Video Coding
TML	Test Model Long-Term
TS-EPDS . . .	Two-step Edge based Partial Distortion Search
UMHS	Unsymmetrical-cross Multi-hexagon Grid Search
VBME	Variable Block size Motion Estimation
VCEG	Video Coding Expert Group
VCD	Video CD
VLC	Variable Length Coding
XP	Extended Profile

Chapter 1

INTRODUCTION

With the widespread adoption of video technologies, video coding has been a hot topic that can effectively handle the redundancies in video data. Digital video has taken the place of traditional analogue video in a wide range of applications due to its compatibility with other types of data (such as voice and text). However, digital video contains huge amount of data and despite the increases in processor speed and disc storage capacity. For example, using a video format of 352×240 pixels with 3 bytes of color data per pixel, playing at 30 frames per second, 7.6 Megabytes of disc space is needed for one second of video and it is only feasible to store around 10 minutes of video in a 4.3 Gigabytes Digital Versatile Disk (DVD). When it is transmitted in real time through the internet, it requires a channel with 60.8 mbps. But the data throughput of consumer DSL services typically ranges from 256 Kb/s to 20 Mbit/s in the direction to the customer (downstream), depending on DSL technology, line conditions, and service-level implementation [1].

Video coding technologies are about reducing and removing redundant video data so that a digital video file can be effectively sent over a network and stored on computer disks. With efficient compression techniques, a significant reduction in file size can be achieved with little or no adverse effect on the visual quality. There are two classes of techniques for image and video compression: lossless coding and lossy coding [2]. Lossless coding techniques compress the image and video data without any loss of information and the compressed data can be decoded exactly the same as original data; however, these techniques obtain a very low compression ratio and result in large files. Consequently, they are appropriate for applications requiring no loss introduced by compression, for example, medical image storage. On the other hand, lossy coding methods sacrifice some image and video quality to achieve a significant decrease in file size and a high compression ratio. Lossy coding techniques are widely used in digital image and video applications due to the high compression ratio provided. The goal of a video compression algorithm is to achieve efficient compression as well as minimizing the distortion introduced by the compression process [3].

1.1 Significance of video compression

Recent advances in digital video technology have fundamentally changed the way visual information is processed, stored and transferred. As other technological components such as semiconductor, microprocessors, digital communication and digital signal processing become mature and available, digital video is replacing the traditional analog video with the emergence of various applications such as High Definition TV (HDTV) , Video CD (VCD), Digital Versatile Disk (DVD), Digital Video Broadcasting (DVB) , and video conferencing. An advantageous property of digital video is that many digital signal processing techniques (e.g., compression, encoding, encryption and error correction) can be directly applied on the video signal which has been digitized. Another reason for the trend of such transition from analog to digital in video technology can be attributed to the wide availability of digital storage devices, computers, digital signal processors and digital communication systems, so that video information is more likely to be processed, stored, and transferred on those digital platforms.

However, video is also one of the most bandwidth-consuming types of information. Even as new technology causes communication channels with ever-increasing bandwidth, compression of the video data is still necessary to achieve more efficient bandwidth utilization. For example, in the terrestrial TV broadcasting, satellite TV and cable TV applications, more high resolution video channels can be delivered within the assigned bandwidth if each video signal is compressed for transmission. In the applications of video transmission over digital data network, compression of the video data is necessary to reduce the impact and burden on the network traffic. In internet streaming or wireless transmission applications, the quality that is considered to be acceptable is much lower. The target bit rates are in the range of 8-384 kbit/s and the spatial and temporal resolution of the video is reduced. Therefore, the development of efficient mode selection scheme for video codec has attracted substantial research interest.

1.2 Video Compression Standard

The adoption of digital video in many applications has been fueled by the development of many video coding standards, which have emerged targeting different application areas.

These standards provide the means needed to achieve inter operability between systems designed by different manufacturers for any given application, hence facilitating the growth of the video market. The International Telecommunications Union, Telecommunications Standardization Sector (ITU-T) is now one of two formal organizations that develop video coding standards-the other being International Standardization Organization / International Electro-technical Commission, Joint Technical Committee 1 (ISO/IEC JTC1). The ITU-T video coding standards are called recommendations, and they are denoted with H.26x (e.g., H.261, H.262, H.263 and H.264). The ISO/IEC standards are denoted with MPEG-x (e.g., MPEG-1, MPEG-2 and MPEG-4).

International study groups, VCEG (Video Coding Experts Group) of ITU-T (International Telecommunication Union—Telecommunication sector), and MPEG (Moving Picture Experts Group) [4] of ISO/IEC, have researched the video coding techniques for various applications of moving pictures since the early 1985s. Since then, ITU-T developed H.261 as the first video coding standard for video conferencing application. MPEG-1 [5] video coding standard was accomplished for storage in compact disk and MPEG-2 [6] (ITU-T adopted it as H.262) standard for digital TV and HDTV as extension of MPEG-1 [7]. Also, for covering the very wide range of applications such as shaped regions of video objects as well as rectangular pictures, MPEG-4 part 2 [8] standard was developed. This includes also natural and synthetic video/audio combinations with interactivity built in. On the other hand, ITU-T developed H.263 [9]-[12] in order to improve the compression performance of H.261, and the base coding model of H.263 was adopted as the core of some parts in MPEG-4 part 2. MPEG-1, -2, and -4 also cover audio coding. Fig.1.1 summarizes the evolution of the ITU-T recommendations and the ISO/IEC MPEG standards.

To provide better compression of video compared to previous standards, H.264/ MPEG-4 part 10 [13] video coding standard was recently developed by the JVT (Joint Video Team) [14] consisting of experts from VCEG and MPEG. H.264 fulfills significant coding efficiency, simple syntax specifications, and seamless integration of video coding into all current protocols and multiplex architectures. Thus, H.264 can support various applications like video broadcasting, video streaming, video conferencing over fixed and wireless networks and over different transport protocols.

H.264 video coding standard has the same basic functional elements as previous standards (MPEG-1, MPEG-2, MPEG-4 part 2, H.261, and H.263) [15], i.e., transform for reduction of spatial correlation, quantization for bit rate control, motion compensated prediction for reduction of temporal correlation, and entropy encoding for reduction of statistical correlation.

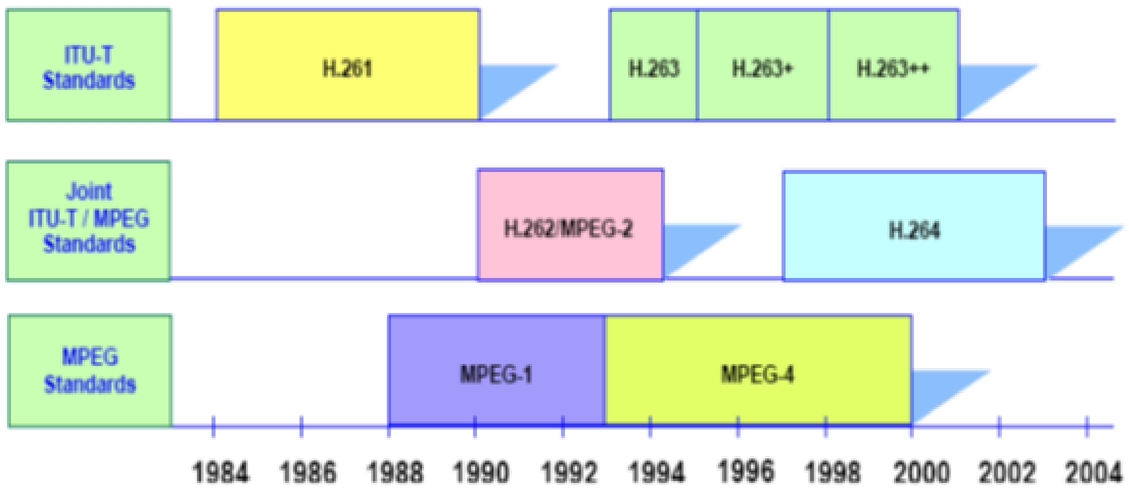


Fig. 1.1 Progression of the ITU-T Recommendations and MPEG standards.

However, to fulfill better coding performance, the important changes in H.264 occur in the details of each functional element by including intra-picture prediction, a new 4×4 integer transform, multiple reference pictures motion estimation, variable block sizes and a quarter pixel precision for motion compensation, a de-blocking filter, and improved entropy coding. It also uses seven different block sizes for motion compensation in the inter-mode and three different block sizes with various spatial direction modes in the intra-frame.

1.3 Statement of the problem

H.264/AVC is the newest international video compression standard. H.264/AVC has been demonstrated to provide significant rate-distortion gains over previous standards, and it is widely accepted as the state-of-the-art in video coding standard [16]-[19]. H.264/AVC has many similar characteristics to previous standards, but some of the main new features are outlined below:

- Up to five reference fames may be used for motion estimation (as opposed to the one or two frames used in previous standards).
- For each 16×16 macro-block, variable block size motion estimation is used [20]. This allows a range of different block sizes for motion compensation — from 16×16 down to 4×4 pixels. Using seven different block sizes and shapes can translate into bit rate savings of more than 15% as compared to using only a 16×16 block size [21]. However, computational complexity of this method is extremely high.

- Motion vectors can be specified to one-quarter pixel accuracy (or one eighth pixel in the case of chrominance components).

- Intra-frame coding is performed using 4×4 blocks, based on a fast integer approximation of the DCT. Spatial prediction within frames is also used to achieve additional de-correlation.

- An adaptive de-blocking filter is used within the motion compensation loop in order to improve picture quality.

- Context-adaptive binary arithmetic coding (CABAC) is employed

All of the above new features improve rate-distortion (RD) performance of the encoder with the expense of extremely high computations. Among the several new features which are introduced by H.264/AVC, the motion estimation (ME) and mode decision process is highly computationally intensive than traditional algorithms. Therefore, the development of efficient algorithms for the ME and mode decision of H.264/AVC is one of the most challenging themes. Table 1.1 shows the total encoding time 100 frames of eight QCIF (176×144) sequences in different video coding standards [22].

It is shown that total encoding time of H.264/AVC baseline profile is around 106 times of H.263+. It is easy to foresee that the computational complexity will further increase dramatically if higher picture resolutions, for instant CIF (352×288) and CCIR-601 (720×480), are required. Therefore, efficient algorithms to reduce the computational complexity of H.264/AVC without compromising the coding efficiency are indispensable for real time implementation.

Table 1.1 Encoding time of different video coding standards [22]

Standard-compliant encoder	Total encoding time (second)
MPEG-1	26.65
H.263	66.37
H.263+	126.70
H.264 Baseline Profile	13387.83
H.264 Main Profile	19264.53
H.264 Extended Profile	20713.22

The main purpose of this dissertation is to develop fast and efficient mode (Intra and Inter) decision algorithms to reduce the complexity of the H.264/AVC encoder, at the same time without degrading the picture quality as compared to original method.

1.4 Objectives of this dissertation

From the background study and after the problem statement, to overcome the drawback of mode selection process in H.264/AVC, the objectives are first identified and then proceed with the development and implementation of fast and efficient mode selection schemes which brought significant contribution in video coding are as follows:

1. To develop efficient mode selection schemes for selecting the right macroblock to determine the best one earlier in latest video coding standard.
2. To reduce the encoder computation time since the tremendous computation of the RD cost can be skipped.
3. To simplify the different parameters selections process that will be very robust for different kind of video sequences.
4. To determine the optimized threshold value for different parameters selection that will be adaptive for different types of video contents.
5. Finally, to perform the experimental validation of the developed scheme with the latest video coding standard H.264/AVC encoder.

1.5 Contribution made in this dissertation

Within H.264/AVC video coding, motion estimation and rate-distortion optimized mode decisions contributes to the largest gain in compression but both of these are also the most computationally intensive parts which limits to real time implementation. In motion estimation, similarities between different video frames are searched and identified; redundant data are then eliminated or minimized to reduce temporal redundancy within a video sequence. Rate-distortion optimized intra/Inter mode decision is also one of the important coding tools of H.264/AVC encoder. Many motion estimation and mode selection scheme have been developed over the last decade; most of these methods come with a complex search flow, extensive computation complexity and with a limited speedup. To overcome the drawback of these in H.264/AVC, some novel and efficient mode selection scheme have been developed which brought significant contribution in video coding and in this approach following contributions has been made.

1. This dissertation proposes a modified DC prediction (MDCP) mode scheme for Intra 4×4 based on the distance between the predicted and reference pixels In H.264/AVC intra coding, DC mode is used to predict regions with no unified direction and the predicted pixel values are same and thus smooth varying regions are not well decorrelated.

2. A new Z-scan scheme is proposed to improve intra coding efficiency for the AVS standard. In proposed method, traditional zigzag scan, horizontal z-scan, and vertical z-scan are used depending on the spatial prediction directions. It is easy to implement the proposed scheme into AVS codec without changing the syntax. The proposed scheme can remarkably reduce bit rates compared with AVS codec using zigzag scan, while the PSNR of video sequences are maintained. Within this scheme, according to the spatial prediction directions, two additional scan table called horizontal Z-scan and vertical Z-scan are used together with normal zigzag scan to organize coefficients.

3. A novel rate-based macroblock classification for fast mode decision in H.264/AVC video coding standard is proposed. The main idea is to classify each macroblock into simple motion or complex motion contents based on the Inter 16×16 mode's residue block bit-rate and then according to the classification different mode searching orders with distinct early termination schemes are employed. This new algorithm is very simple for both hardware and software implementations without extra computational module. To speed up the intra mode decision, a new fast Intra 4×4 mode selection algorithm is also proposed by choosing most likely modes using the low complexity SATD cost as screening function. It is demonstrated by experimental results that the proposed rate-based fast algorithm can reduce significant encoding time with negligible degradation in the rate-distortion performance. A further speedup technique is also introduced in this scheme. As the encoding of the Inter 16×16 mode is part of the rate-distortion optimization mode selection process, thus no additional operation module is required for the proposed fast algorithm.

4. A novel relative sum of absolute difference (RSAD) based inter mode decision algorithm is proposed for H.264/AVC video standard. The main idea is to determine the best inter mode based on RSAD cost instead of the rate-distortion cost. This approach can avoid most of the computationally intensive processes in the H.264/AVC mode decision, such as integer transform, quantization, variable length coding and

pixel-reconstruction. Besides, it can solve the problem of SAD-based cost function for H.264/AVC mode decision in biasing to the smaller block size modes, as they are normally achieving higher prediction accuracy. The proposed algorithm can reduce significant encoding time with negligible degradation in the rate-distortion performance. Complexity savings are also achieved by processing the macroblocks that are predicted as "skipped". The proposed algorithm is also combined with SKIP mode early termination algorithm for further time saving and computation reduction. The enhanced algorithm with SKIP mode early termination technique could save more encoding time with only a little rate-distortion degradation. The proposed scheme can avoid rate-distortion cost computation while maintaining high rate-distortion performance of the H.264/AVC codec.

5. A new probability based fast inter mode decision algorithm is proposed for H.264/ AVC video standard. The main idea is to classify the inter modes into large and small modes category and determine which one is suitable for a macroblock (MB). Depending on this, the probability of each large mode corresponding to the initial Hadamard $SAD_{16\times16}$ cost or the probability of each small mode corresponding to the initial Hadamard $SAD_{s8\times8}$ cost is determined. Then the best mode can be obtained from their probability characteristics. Thus, it is possible to determine the best inter mode only by calculating the Hadamard Transform (H-SAD) cost instead of rate distortion cost. Moreover, it avails the platform to determine the best mode without comparing the cost of each mode to other modes. As a result, this approach can avoid most of the complex computation processes like quantization, variable length coding, pixel reconstruction etc. The proposed based scheme can decide the best mode without calculating the rate distortion cost maintaining a high rate distortion performance of the H.264/ AVC codec.

6. Finally, a new approach of search point reduction, ranking and performance analysis of commonly used motion estimation (ME) algorithms is introduced for H.264/AVC encoder. Since number of searching point is not a direct measure of hardware performance, a cost function of accessing data in memory is defined and then, an adjustments is given on the existing ME algorithms to minimize the problem of memory access by reordering the sequence of searching points, where the number of searching points and MAE/PSNR remains the same as the original algorithms. In addition to this, the ranking of memory access cost for small and high motion video with lower computational complexity and similar quality is maintained is by the experimental evaluation.

These fast and efficient algorithms are likely to be useful in implementing real-time H.264/AVC standard encoders in computationally constrained environments such as low-power mobile devices and general purpose computers. The developed fast mode selection schemes for H.264/AVC could be applied on digital video recorder, video conferencing system, 3rd generation mobile phone, remote video security system, video entertainment system, etc. This may greatly help to develop the fast data transmission for real time video processing. So, with the help of this dissertation, the area of video coding will be explored and our country will be enriched with new technology as well as it may lead to carry out the research on video technology in Bangladesh or anywhere in the world.

1.6 Organization of the dissertation

The remainder of this dissertation is organized as follows:

- In chapter 2, a more detailed survey on the video coding standard H.264/AVC is provided.
- In chapter 3, a modified intra prediction scheme of H.264/AVC is proposed.
- A modified Z-scanning scheme for directional spatial prediction of AVS Intra coding is proposed in chapter 4.
- Chapter 5 deals with a rate-based macroblock classification for fast mode decision in H.264/AVC video coding.
- Relative SAD-based inter mode decision algorithm for H.264/AVC Video Standard is discussed in chapter 6.
- Probability based fast inter mode decision algorithm for H.264/AVC Video Standard is described in chapter 7.
- A new approach of search point reduction, ranking and performance analysis of commonly used motion estimation (ME) algorithms for H.264/AVC encoder is introduced in chapter 8.

Finally, conclusions and future research directions of the is dissertation are given in chapter 9.

Chapter 2

OVERVIEW OF THE LATEST VIDEO CODING H.264/AVC STANDARD

To provide better compression of video compared to previous standards, H.264/ AVC [23] video coding standard was recently developed by the JVT (Joint Video Team) consisting of experts from VCEG and MPEG. H.264/AVC fulfills significant coding efficiency, simple syntax specifications, and seamless integration of video coding into all current protocols and multiplex architectures. Thus, H.264/AVC can support various applications like video broadcasting, video streaming, video conferencing over fixed, wireless networks and over different transport protocols [24]. In this chapter, the main features of the H.264/AVC are summarized as the detail background study in the field of latest video technology standard tools.

2.1 History

In early 1998 the Video Coding Experts Group (VCEG - ITU-T SG16 Q.6) issued a call for proposals on a project called H.26L [25], with the target to double the coding efficiency (which means halving the bit rate necessary for a given level of fidelity) in comparison to any other existing video coding standards for a broad variety of applications. The first draft (Test Model Long-Term TML-1) design for that new standard was adopted in August 1999 [26]. In December 2001, the Moving Pictures Experts Group (MPEG) joined the standardization process in the Joint Video Team (JVT) to finalize the joint recommendation/standard H.264/AVC. The development started off with the Joint Test Model JM-1 [27]. The JM-1 departed from TML-9 [28]. The status of a Final Draft International Standard was reached in March 2003 [29]. In June 2004, the Fidelity range extensions (FRExt) project was finalized [30]. From

January 2005 to November 2007, the JVT was working on an extension of H.264/AVC towards scalability by an Annex (G) called Scalable Video Coding (SVC). From July 2006 to November 2009, the JVT worked on Multiview Video Coding (MVC), an extension of H.264/AVC towards free viewpoint television and 3D television.

2.2 Terminology

Some of the important terminology adopted in the H.264/AVC standard is as follows.

2.2.1 Pictures, frames and fields

A coded video sequence in H.264/AVC consists of a sequence of coded pictures. A coded picture can be represented either an entire frame or a single field [18]. There are two types of video supported in H.264/AVC; interlaced and progressive [30]. In interlaced video, each video frame is divided into two fields. The first field consists of odd numbered picture line (1, 3, 5,..........…) and second field is formed by taking the even lines (2, 4, 6,. . . . .). A progressive video frame is not divided into any sections; the entire picture is coded and transmitted as one unit.

2.2.2 YC_bC_r Color space and 4:2:0 sampling

The human visual system (HVS) is less sensitive to color than to luminance (brightness) [18]. Video transmission systems can be designed to take advantage of this. The video color space used by H.264/AVC separates a color representation into three components called Y,C_b and C_r. Component Y is called luma, and it represents brightness. The two chroma components C_b and C_r represent the extent to which the color deviates from gray toward blue and red, respectively. In H.264/AVC as in prior standards, a Y,C_b, C_r color space is used to reduce the sampling resolution of the C_b and C_r chroma information [17].

2.2.3 4:4:4 resolution

4:4:4 sampling means that the three components (Y,C_b and C_r) have the same resolution and hence a sample of each component exists at every pixel position. In the popular 4:2:0 sampling format, C_b and C_r each have half the horizontal and vertical resolution of Y. Because each color difference component contains one quarter of the number of samples in the Y component, 4:2:0 Y,C_b, C_r video requires exactly half as many samples as 4:4:4 video.

H.264 supports coding and decoding of 4:2:0 progressive or interlaced video and the default sampling format is 4:2:0 progressive frames. Example:

- **Image resolution:** 720×576 pixels, Y resolution: 720×576 samples, each represented with eight bits.
- **4:4:4 C_b, C_r resolution:** 720×576 samples, each eight bits, total number of bits: $720 \times 576 \times 8 \times 3 = 9953280$ bits
- **4:2:0 C_b, C_r resolution:** 360×288 samples, each eight bits, total number of bits: $(720 \times 576 \times 8) + (360 \times 288 \times 8 \times 2) = 4\,976\,640$ bits
  The 4:2:0 version requires half as many bits as the 4:4:4 version.

2.2.4 Macroblock and slices

H.264/AVC uses block based coding schemes. In these schemes, the pictures are subdivided into smaller units called macroblocks that are processed one by one, both by the decoder and the encoder. A macroblock (MB) contains coded data corresponding to a 16×16 sample region of the video frame (16×16 luma samples, 8×8 C_b and 8×8 C_r samples). MBs are numbered (addressed) in raster scan order within a frame.

A video picture is coded as one or more slices, each containing an integral number of MBs from 1 (1 MB per slice) to the total number of MBs in a picture (1 slice per picture). The number of MBs per slice need not be constant within a picture. A typical structure of macroblocks and slices the image sequence of I, P and B are shown in Fig. 2.1 and Fig. 2.2. There is minimal inter-dependency between coded slices which can help to limit the propagation of errors. There are five types of coded slice and a coded picture may be composed of different types of slices [18].

- I (Intra) slice: Contains only I MBs. Each block or all MBs is predicted from previously coded data within the same slice.
- P (Predicted) slice: In addition to the coding types of the I slice, some MBs of the P slice can also be coded using inter prediction with at most one motion compensated prediction signal per prediction block.
- B (Bi-predictive): In addition to the coding types available in a P slice, some MBs of the B slice can also be coded using inter prediction with two motion compensated prediction signals per prediction block.

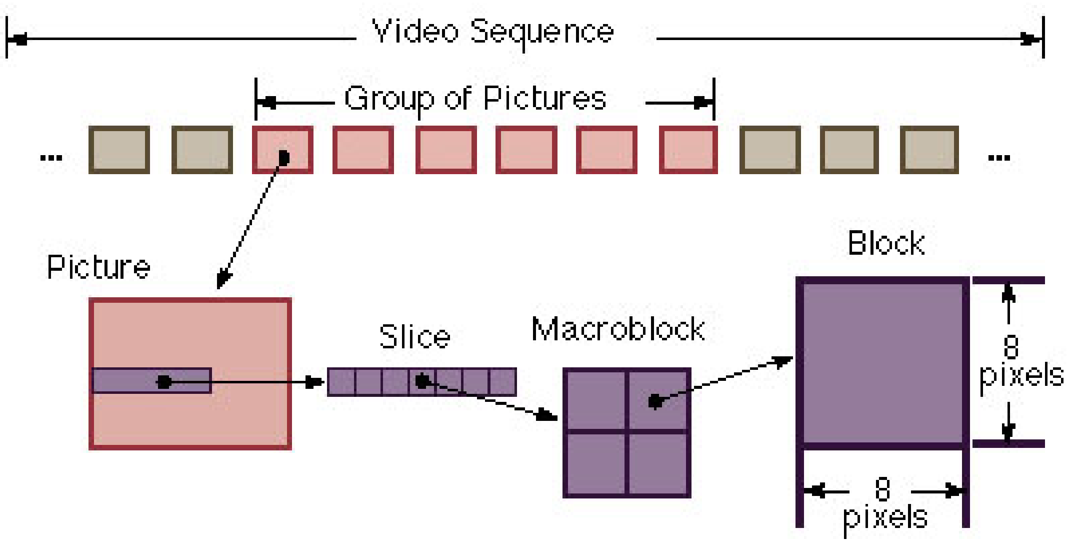


Fig. 2.1 Macroblock and Slices.

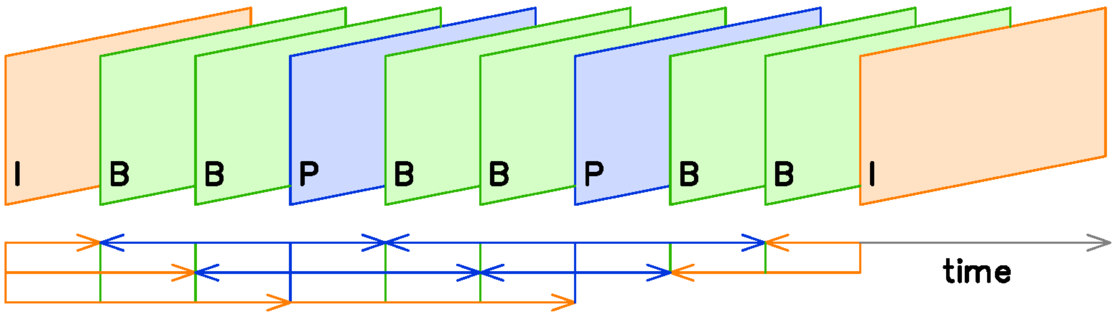


Fig. 2.2 IPB images sequence.

- SP (Switching P): A so-called switching P slice that is coded such that efficient switching between different precoded pictures becomes possible [31].
- SI (Switching I): A so-called switching I slice that allows an exact match of a MB in an SP slice for random access and error recovery purposes [31].

2.3 H.264/AVC Profiles

While H.264/ AVC standard contains a rich set of video coding tools, not all the coding tools are required for all applications. For example, error resilience tools may not be needed for video stored on a compact disk or on networks with very few errors [32]. Therefore, the standard defines subsets of coding tools intended for different classes of applications. These subsets are called Profiles. There are three Profiles in the first version: Baseline (BP), Main (MP), and Extended (XP) [33]. Baseline Profile is to be applicable to real-time conversational services such as video conferencing and videophone. Main Profile is designed for digital storage media and television broadcasting. Table 2.1 shows the features supports in different profiles. Extended Profile is aimed at multimedia services over Internet. Also there are four High Profiles defined in the fidelity range extensions [34] for applications such as content-contribution, content-distribution, and studio editing and post-processing: High (Hi), High 10 (Hi10), High 4:2:2 (Hi422), and High 4:4:4 (Hi444). High Profile is to support the 8-bit video with 4:2:0 sampling for applications using high resolution. High 10 Profile is to support the 4:2:0 sampling with up to 10 bits of representation accuracy per sample. High 4:2:2 Profile is to support up to 4:2:2 chroma sampling and up to 10 bits per sample. High 4:4:4 Profile is to support up to 4:4:4 chroma sampling, up to 12 bits per sample, and integer residual color transform for coding RGB signal.

2.4 Block diagram of H.264/AVC

H.264/AVC is a recently completed video compression standard jointly developed by the ITU-T VCEG and the ISO/IEC MPEG standard committee [6]. The standard promises much higher compression than that possible with earlier standard. However the increase in compression efficiency comes at the cost of substantial increase in complexity, often quoted as factor of four for the decoder, whereas coding complexity may be as high as factor of nine over MPEG-2 [6]. New and advanced techniques are introduced in this new standard, such as intra prediction for I-frame encoding, multi-frames inter prediction, small block-size transform coding, context-adaptive arithmetic entropy coding, de-blocking filtering, etc.

These advanced techniques make this new standard provides approximately 50% bit rate saving for equivalent perceptual quality relative to the performance of prior standards. In this chapter, we will focus on the H.264/AVC which employs many new advanced features to improve the compression efficiency compared with previous video standards.

Table 2.1 Features of different profiles of H.264

Feature	BP	MP	XP	Hi	Hi10	Hi422	Hi444
B slices	No	Yes	Yes	Yes	Yes	Yes	Yes
Flexible macroblock ordering(FMO)	Yes	No	Yes	No	No	No	No
Data partitioning	No	No	Yes	No	No	No	No
Interlaced	No	Yes	Yes	Yes	Yes	Yes	Yes
CABAC entropy coding	No	Yes	No	Yes	Yes	Yes	Yes
8×8 vs. 4×4 transform adaptivity	No	No	No	Yes	Yes	Yes	Yes
Quantization scaling matrices	No	No	No	Yes	Yes	Yes	Yes
Separate Cb and Cr QP control	No	No	No	Yes	Yes	Yes	Yes
Monochrome (4:0:0)	No	No	No	Yes	Yes	Yes	Yes
Chroma formats	4:2:0	4:2:0	4:2:0	4:2:0	4:2:0	4:2:0/4:2:2	4:2:0/4:2:2/4:4:4
Simple depths(bits)	8	8	8	8	8 to 10	8 to 10	8 to 14
Separate color plane coding	No	No	No	No	No	No	Yes
Predictive lossless coding	No	No	No	No	No	No	Yes

H.264/AVC used hybrid video scheme structure[35]. Fig. 2.3 shows the block diagram of H.264/AVC codec. An input frame F_n is processed in units of a macroblock. Each macroblock is coded in intra or inter mode and, for each block in the macroblock, a predicted block P is formed from samples in the current slice that have previously encoded, decoded and reconstructed (uF_n'). In inter mode, prediction is formed by motion-compensated prediction from previous reconstructed frames. Motion estimation and motion compensation [36] [37] play very important roles in the hybrid coding scheme, since they can greatly reduce the temporal redundancy between adjacent video frames. Generally speaking, the adjacent pictures always share much similar MBs; thus, the current MB can be presented by the other one in the previous frame with very little difference. As illustrated in Fig. 2.4 there are two adjacent frames from the video “reading”.

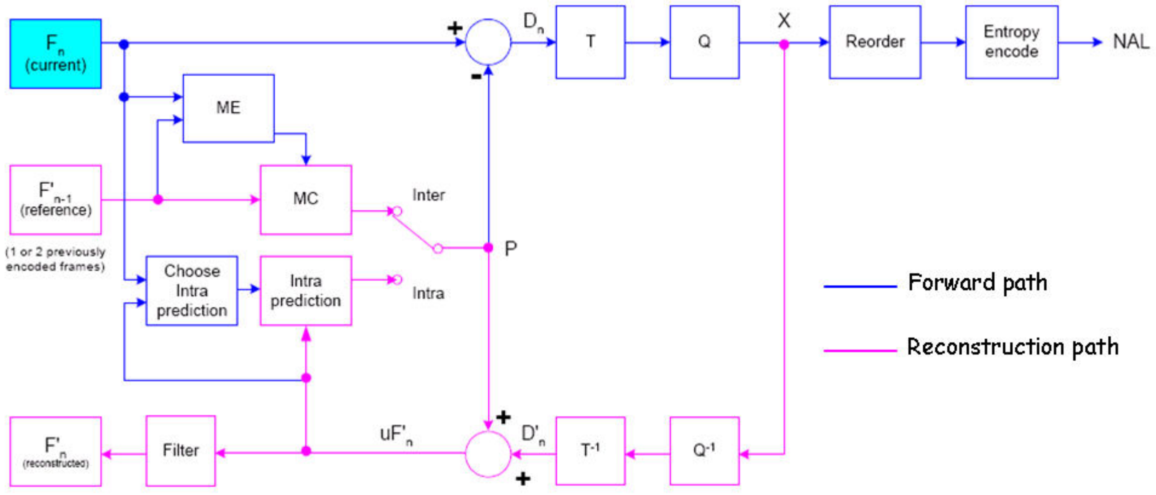


Fig. 2.3 Block diagram of H.264/AVC encoder.

Though the two pictures are very similar, if the frame 1 is encoded with reference to frame 2, a large amount of residual data will be left without motion compensation, as shown in Fig. 2.4 (c). However, if the panning motion is compensated by finding the displacement with motion estimation, the remaining residual data will be very little, as shown in Fig. 2.4 (d). For this reason, the necessary information to be coded is substantially reduced, and the huge video data can be effectively compressed. The predicted block is subtracted from the current block to produce residual block D_n. The residual data is then discrete cosine transformed (DCT) [38], [39] and quantized for lossy compression to give X which contains the quantized DCT coefficients [40]. The quantized transform coefficients are recorded and then entropy coded.

The encoder decodes a macroblock to provide a reference for further predictions. The coefficients X are scaled (Q^{-1}) and inverse transformedT^{-1} to produce a difference blockD'_n. The predicted block P is added to D'_n to create a reconstructed block uF'_n. A filter is applied to reduce the effect of blocking distortion and reconstructed reference picture is created from a series of block F'_n.

2.5 Intra Prediction

Intra coding refers to the case where only spatial redundancies within a video picture are exploited. The resulting frame is referred to as an I-frame. I-frames are typically encoded by directly applying the transform to the different macroblocks (MBs) in the frame. Consequently, encoded I-pictures are large in size since a large amount of information is usually present in the frame, and no temporal information is used as part of the encoding process.

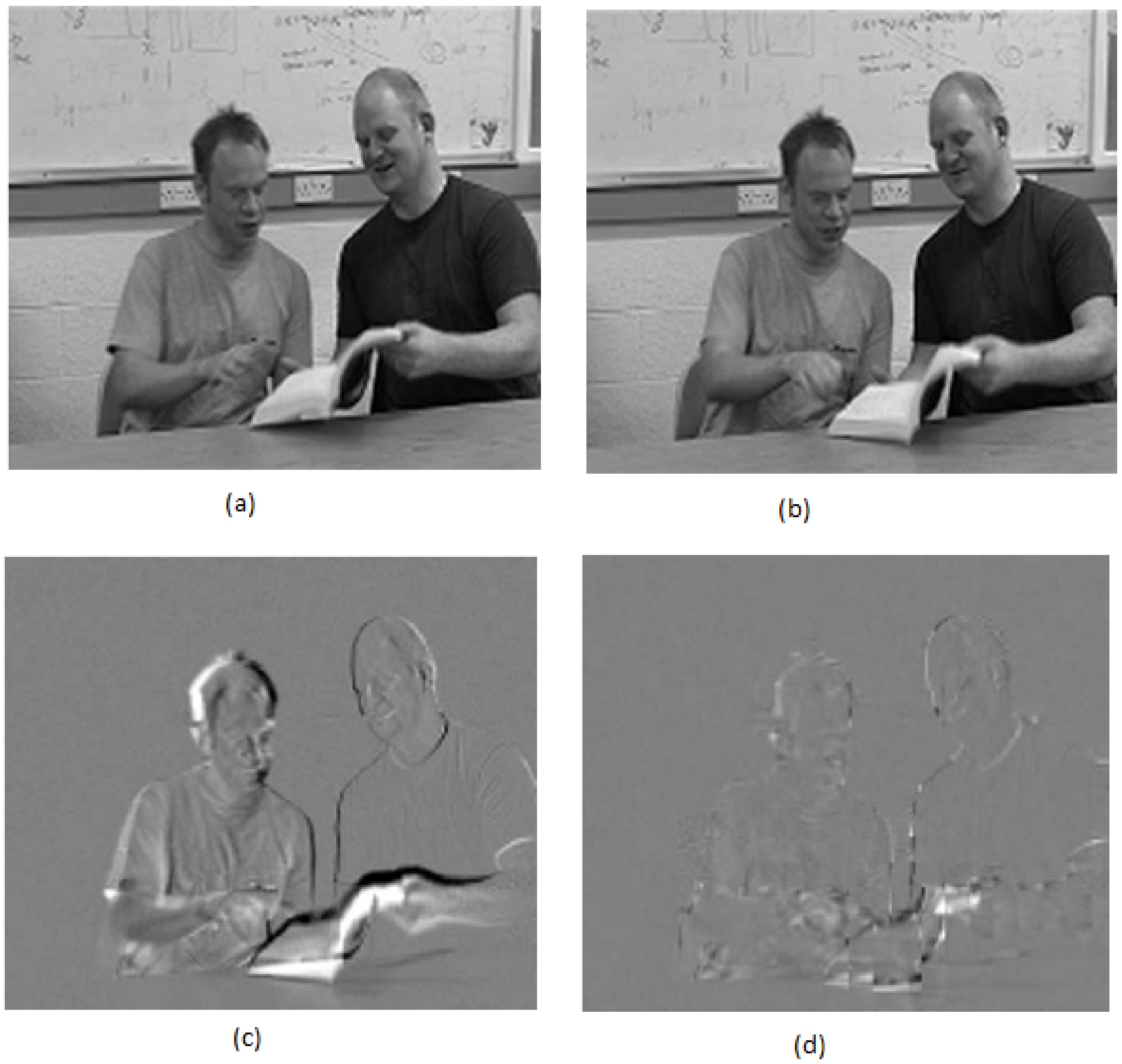


Fig. 2.4 (a) Frame 1, (b) Frame 2, (c) The residual picture by subtracting the frame 1 from frame 2 without motion compensation, (d) The residual picture by subtracting the frame 1 from frame 2 with motion compensation.

In order to increase the efficiency of the intra coding process inH.264/AVC, spatial correlation between adjacent MBs in a given frame is exploited [21]. The idea is based on the observation that adjacent MBs tend to have. Therefore, as a first step in the encoding process for a given MB, one may predict the MB of interest from the surrounding MBs (typically the ones located on top and to the left of the MB of interest, since those MBs would have already been encoded). The difference between the actual MB and its prediction is then coded, which results in fewer bits to represent the MB of interest as compared to when applying the transform directly to the MB itself.

In contrast to some previous standards (namely H.263+ and MPEG-4 Visual), where intra prediction conducted in the transform domain, intra prediction in H.264/AVC is always conducted in spatial domain, by referring to neighboring samples has been of previously coded blocks which are to the left and/or above the block to be predicted. This may incur error propagation in environments with transmission errors that propagate due to motion compensation into inter-coded macroblocks [12]. Therefore, a constrained intra coding mode can be signaled that allows prediction only from intra-coded neighboring macroblocks.

For the luminance (luma) samples, intra prediction may be formed for each 4×4 block or for a 16×16 macroblock. There are a total of 9 optional prediction modes for each 4×4 luma block; four optional modes for a 16×16 luma block. The latest H.264 standard also defines 8×8 block and also has nine prediction modes which are the same as those modes used in 4×4 block. Similarly for chroma 8×8 block, another four prediction direction is used.

2.5.1 Intra 4×4 Prediction

In case of 4×4 luminance block, the prediction block is defined using neighboring pixels of reconstructed blocks. The prediction block is calculated based on the samples labeled A-M as shown in Fig. 2.5. For example, mode 2 is called DC prediction in which all pixels (labeled a to p) are predicted by (A+B+C+D+I+J+K+L)/8.

The mode 0 specifies the vertical prediction mode in which pixels (labeled a, e, i and m) are predicted from A, and the pixels (labeled b, f, j and n) are predicted from B, and so on. The remaining modes are defined similarly according to the different directions as shown in Fig. 2.6 and Table 2.2. Note that in some cases, not all of the samples above and to the left are available within the current slice: in order to preserve independent decoding of slices, only samples within the current slice are available for prediction.

M	A	B	C	D	E	F	G	H
I	a	b	c	d				
J	e	f	g	h				
K	i	j	k	l				
L	m	n	o	p				

Fig. 2.5 Prediction samples of a 4 × 4 block.

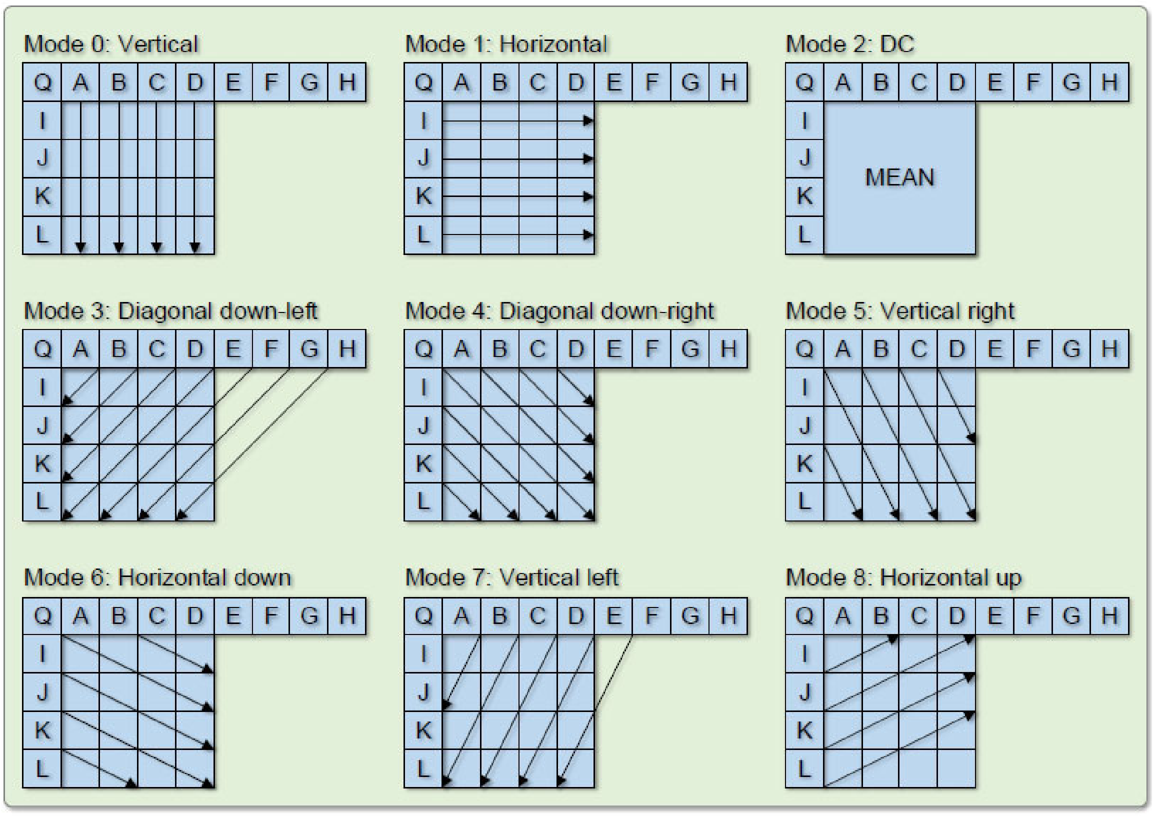


Fig. 2.6 Nine prediction mode of a 4 × 4 block.

DC prediction (mode 2) is modified depending on which samples A-M are available; the other modes (mode 1-8) may only be used if all of the required prediction samples are available The encoder may select the prediction mode for each block that minimizes the residual between the block to be encoded and its prediction. For example, in the case where Mode 3 (Diagonal-Down-Left prediction) is chosen, the values of a to p are given as follows:

- a is equal to (A+2B+C+2)/4
- b, e are equal to (B+2C+D+2)/4
- c, f, i are equal to (C+2D+E+2)/4
- d, g, j, m are equal to (D+2E+F+2)/4
- h, k, n are equal to (E+2F+G+2)/4
- l, o are equal to (F+2G+H+2)/4, and
- p is equal to (G+3H+2)/4.

Table 2.2 Nine intra 4×4 prediction modes

Mode 0 (Vertical)	The upper samples A, B, C, D are extrapolated vertically.
Mode 1 (Horizontal)	The left samples I, J, K, L are extrapolated horizontally.
Mode 2 (DC)	All samples in P are predicted by the mean of samples A . . . D and I . . . L.
Mode 3 (Diagonal down-left)	The samples are interpolated at a 45^0 angle between lower-left and upper-right.
Mode 4 (Diagonal down-right)	The samples are extrapolated at a 45^0 angle down and to the right.
Mode 5 (Vertical right)	Extrapolation at an angle of approximately 26.6^0 to the left of vertical (width/height = 1/2).
Mode 6 (Horizontal down)	Extrapolation at an angle of approximately 26.6^0 below horizontal.
Mode 7 (Vertical left)	Extrapolation (or interpolation) at an angle of approximately 26.6^0 to the right of vertical.
Mode 8 (Horizontal up)	Interpolation at an angle of approximately 26.6^0 above horizontal.

2.5.2 Intra 8×8 Prediction

Similar to intra 4×4 block, 8×8 luma block also has nine prediction mode based on the direction of Fig. 2.6. For prediction of each 8×8 luma block, one mode is selected from the 9 modes, similar to the 4×4 intra-block prediction.

2.5.3 Intra 16×16 Prediction

As an alternative to the 4×4 luma modes described above, the entire 16×16 luma component of a macroblock may be predicted. Four modes are Mode 0 (vertical), Mode 1 (horizontal), Mode 2 (DC) and Mode 3 (Plane). Fig. 2.7 shows the four modes and their direction. Table 2.2 describes the calculation of four modes.

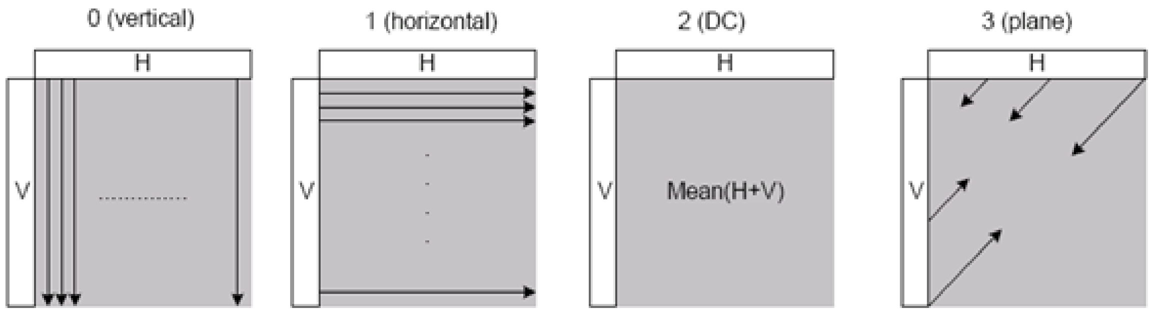


Fig. 2.7 Intra 16×16 prediction modes.

Table 2.3 Four intra 16×16 prediction modes

Mode 0 (Vertical)	Extrapolation from upper samples (H).
Mode 1 (Horizontal)	Extrapolation from left samples (V).
Mode 2 (DC)	Mean of upper and left-hand samples (H + V).
Mode 3 (Plane)	A linear ‘plane’ function is fitted to the upper and left-hand samples H and V. This works well in areas of smoothly-varying luminance.

2.5.4 Intra Chroma Prediction

Each chroma component of a macroblock is predicted from chroma samples above and/or to the left that have previously been encoded and reconstructed. The chroma prediction is defined for three possible block sizes, 8×8 chroma in 4:2:0 format, 8×16 chroma in 4:2:2

format, and 16×16 chroma in 4:4:4 format [29]. The four prediction modes for all of these cases are very similar to the 16×16 luma prediction modes, except that the order of mode numbers is different: mode 0 (DC), mode 1 (horizontal), mode 2 (vertical), and mode 3 (plane).

2.6 Inter Prediction

Inter-prediction and coding is based on using motion estimation and compensation to take advantage of the temporal redundancies that exist between successive frames, hence, providing very efficient coding of video sequences. When a selected reference frame(s) for motion estimation is a previously encoded frame(s), the frame to be encoded is referred to as a P-picture. When both a previously encoded frame and a future frame are chosen as reference frames, then the frame to be encoded is referred to as a B-picture.

Motion estimation in H.264 supports most of the key features adopted in earlier video standards, but its efficiency is improved through added flexibility and functionality. In addition to supporting P-pictures (with single and multiple reference frames) and B-pictures, H.264 supports a new inter-stream transitional picture called an SP-picture. The inclusion of SP-pictures in a bit stream enables efficient switching between bit streams with similar content encoded at different bit rates, as well as random access and fast playback modes. As depicted in Fig. 2.8, the pictures in (a) and (b) are the frame 70 and 71 captured from the sequence ‘Stefan’. To observe the two pictures carefully, we can see that the frame 71 has a small amount of shift to the left due to the camera panning. Though the two pictures are very similar if the frame 71 is encoded with reference to frame 70, a large amount of residual data will be left without motion compensation, as shown in Fig. 2.8 (c). However, if the panning motion is compensated by finding the displacement with motion estimation, the remaining residual data will be very little, as shown in Fig. 2.8 (d). For this reason, the necessary information to be coded is substantially reduced, and the huge volume of video data can be effectively compressed in a very high compression ratio.

2.7 Basic Motion Estimation

Motion estimation is a procedure to locate an object of a current frame from a reference frame. The object size can be as small as a pixel, or as large as a frame, but typically a rectangular block of medium size is used. The shift of the object is basically induced by the motion field which can be estimated by the information in spatial and temporal domains, such as the variances of illumination, the orientation of edges, the distribution of colors,

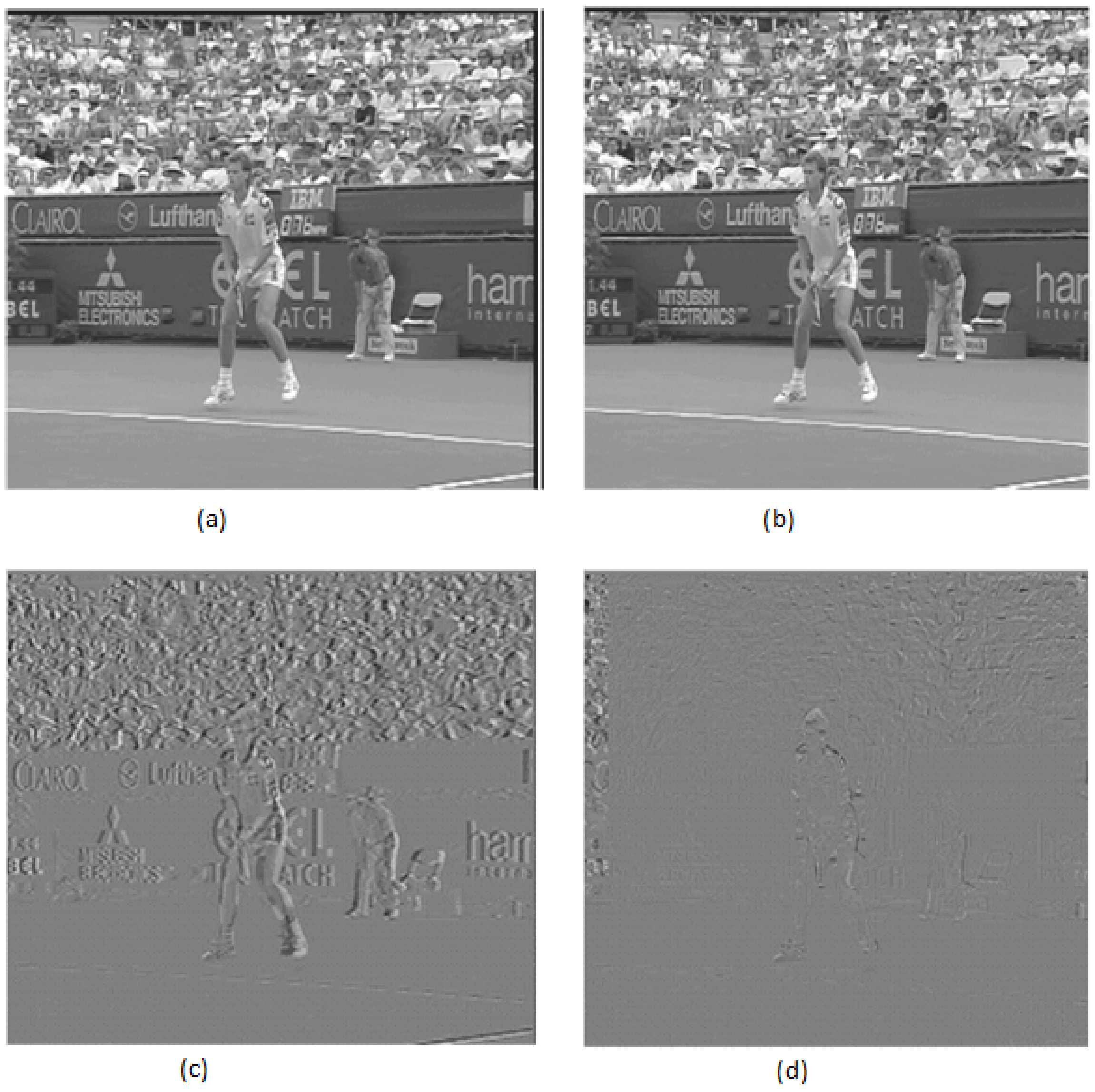


Fig. 2.8 The luminance component of ‘Stefan’ at (a) frame 70, and (b) frame 71. The residual pictures by subtracting the frame 70 from frame 71 (c) without motion compensation, and (d) with motion compensation.

and so on [20]. In general, there are some basic assumptions that most motion estimation algorithms count on:

1. the illumination is constant along the motion path; and
2. the occlusion problem is not present.

These two assumptions confine the complex interactions between motion and illumination to a simple model. The former ignores the problem of illumination changing over time that causes optical flow but not necessary for motion. The latter neglects the typical problems of scene change and uncovered background in which the optical flow is interrupted and not exist for reference. Although the simplified model is not perfect for real-world video contents, the assumptions still hold in most cases.

2.7.1 Block Based Motion Estimation

Block based motion estimation is the most popular and practical motion estimation method in video coding. Standards like the H.26X series and the MPEG series use block based motion estimation. Fig. 2.9 shows how it works. Each frame is divided into square blocks. For each block in the current frame, a search is performed on the reference frame to find a matching based on a block distortion measure (BDM). One of the most popular BDMs is the sum of absolute differences (SAD) between current block and candidate block.

The motion vector (MV) is the displacement from the current block to the best-matched block in the reference frame. Usually a search window is defined to confine the search. Suppose a MB has size $N \times N$ pixels and the maximum allowable displacement of a motion vector (MV) is ± w pixels in both horizontal and vertical directions, there are $(2w+1)^2$ possible candidate blocks inside the search window. Fig. 2.10 shows a search point inside a search window.

A matching between the current MB and one of the candidate MBs is referred as a point being searched in the search window. If all the points in a search window are searched, the finding of a global minimum point is guaranteed. The MV pointing to this point is the optimum MV because it provides the optimum block distortion measure (BDM). This is the simplest block matching algorithm (BMA) and is named Full Search (FS) or exhaustive search. To calculate the BDM of 1 search point using Sum of Absolute Differences (SAD), for a MB of size 16×16 pixels, $3 \times 16 \times 16 - 1 = 767$ operations areneeded (subtract, absolute, and add for each pixel). If FS is used to search all the points in a search window of size $\pm$ 7 pixels, total of $(7 \times 2 + 1)^2 \times 767 = 172575$ operations will be needed for one

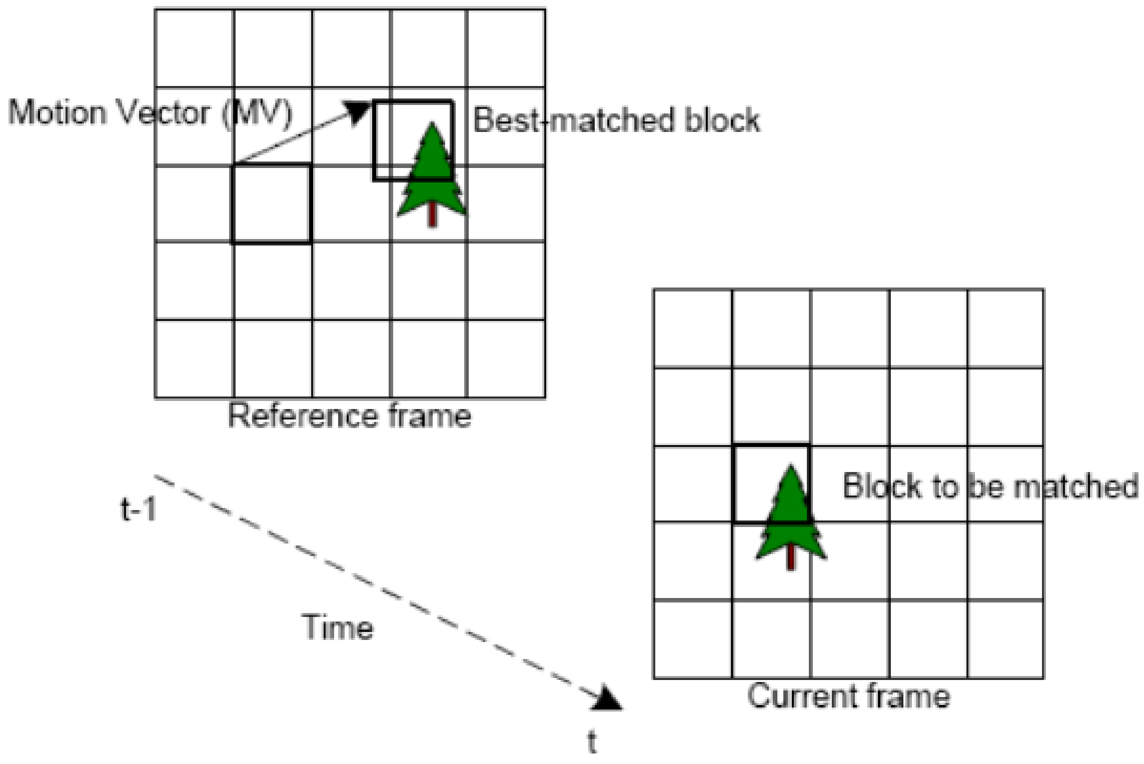


Fig. 2.9 Block matching motion estimation.

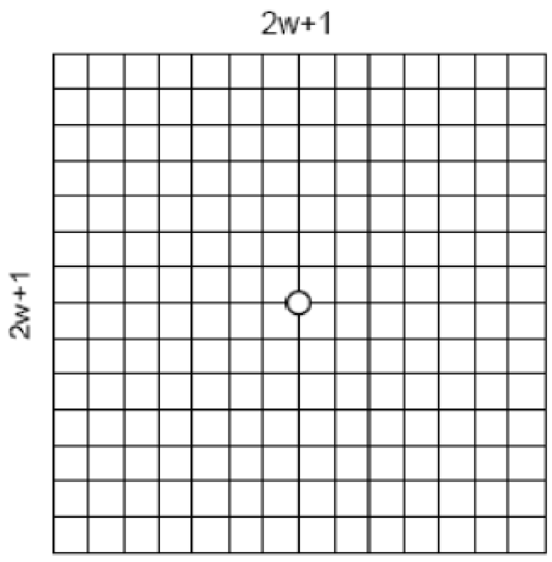


Fig. 2.10 A search point in a search window.

single MB. For newer video coding standards such as the H.264/AVC, which uses variable block-size encoding and multiple reference frames, the number of operations for motion estimation will be even larger.

2.7.2 Variable Block Size Motion Compensation

In order to best represent the motion information, H.264/AVC allows partitioning a macroblock (MB) into several blocks with variable block size, ranging from 16 pixels to 4 pixels in each dimension. For example, one MB of size. 16×16 may be kept as is, decomposed into two rectangular blocks of size 8×16 or 16×8, or decomposed into four square blocks of size 8×8. If the last case is chosen (i.e. four 8×8 blocks), each of the four 8×8 blocks can be further split to result in more sub-macroblocks. There are four choices again, i.e. 8×8, 8×4, 4×8 and 4×4.

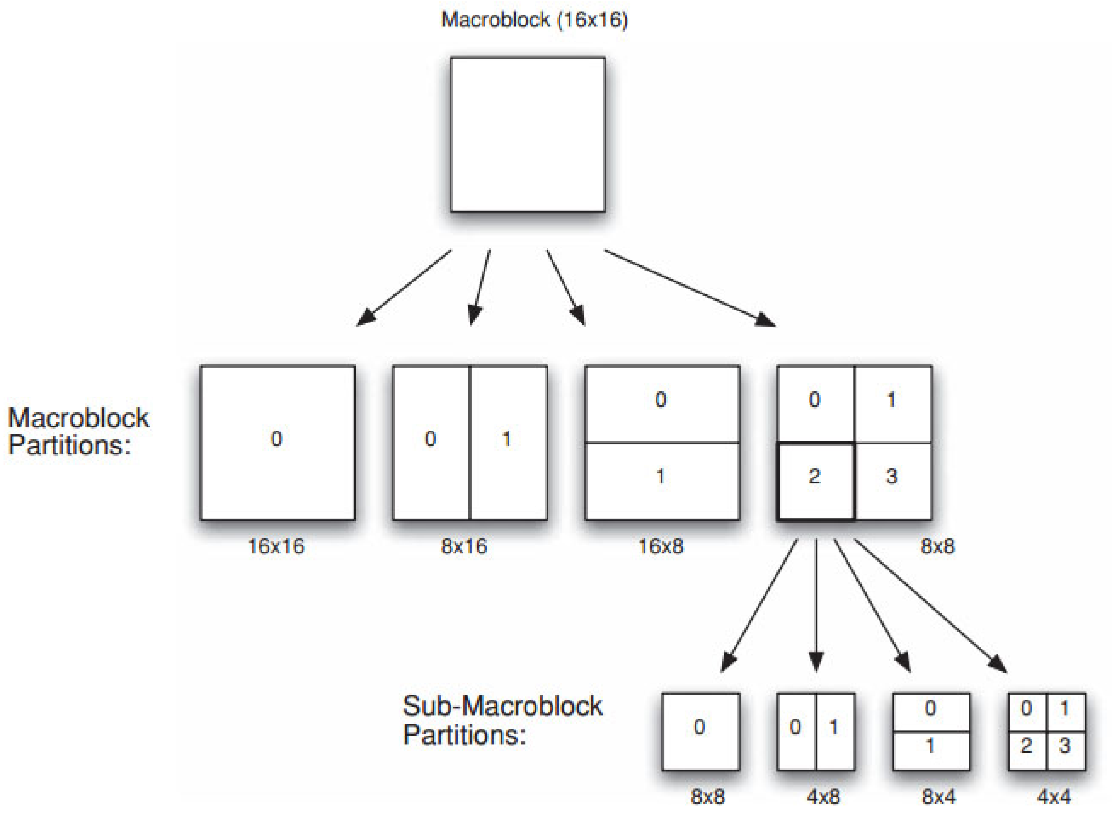


Fig. 2.11 Block sizes for motion estimation of H.264/AVC.

The possible modes of different block sizes are shown in Fig. 2.11. Each block with reduced size can have its individual motion vectors to estimate the local motion at a finer granularity. Though such finer block sizes incur overhead such as extra computation for searching and extra bits for coding the motion vectors, they allow more accurate prediction in the motion compensation process and consequently the residual errors can be considerably reduced, which are usually favorable for the final RD performance.

2.7.3 Sub-Pixel Motion Estimation

The inter-prediction process can form segmentations for motion representation as small as 4×4 luma samples in size, using motion vector accuracy of one-quarter of the luma sample. Sub-pel motion compensation can provide significantly better compression performance than integer-pel compensation, at the expense of increased complexity. Quarter-pel accuracy outperforms half-pel accuracy. Especially, sub-pel accuracy would increase the coding efficiency at high bit rates and high video resolutions. In the luma component, the sub-pel samples at half-pel positions are generated first and are interpolated from neighboring integer pel samples using a 6-tap FIR filter with weights $(1, -5, 20, 20, -5, 1)/32$. Once all the half-pel samples are available, each quarter-pel sample is produced using bilinear interpolation between neighboring half- or integer-pel samples. For 4:2:0 video source sampling, 1/8 pel samples are required in the chroma components (corresponding to 1/4 pel samples in the luma). These samples are interpolated (linear interpolation) between integer-pel chroma samples. Sub-pel motion vectors are encoded differentially with respect to predicted values formed from nearby encoded motion vectors. Detail of sub-pel motion estimation is found in reference [17] and [18].

2.7.4 Multiple Reference Picture Motion Compensation

As the name implies, this concept uses more than one reference frame for prediction. In H.264/AVC, each MB can be predicted using any previously decoded frame in the sequence, which enlarged the search space two to five times. This new feature is very effective for inter frame prediction of the following cases:

1. Motion that is periodic in nature. For example, a flying bird with its wings going up and down. The wings are best predicted from a picture where they are in similar position, which is not necessarily the preceding picture.

2. Alternating camera angles that switch back and forth between two different scenes.

3. Occlusions: once an object is made visible after occlusion, it is beneficial to do prediction from the frame where the object was last visible.

2.7.5 Motion Vector Prediction

Since the encoder and decoder both have access to the same information about the previous motion vectors, the encoder can take advantage of this to further reduce the dynamic range of the motion vector it sends. Encoding a motion vector for each partition can cost a significant number of bits, especially if small partition sizes are chosen. Motion vectors for neighboring partitions are often highly correlated and so each motion vector is predicted from vectors of nearby, previously coded partitions. A predicted vector, MV_p, is formed based on previously calculated motion vectors and motion vector difference (MVD), the difference between the current vector and the predicted vector, is encoded and transmitted. The method of forming the prediction MV_p depends on the motion compensation partition size and on the availability of nearby vectors [17].

2.7.6 Rate-Distortion Optimized Motion Estimation

Earlier encoders typically computed the sum of absolute differences (SAD) between the current block and candidate blocks and selected simply the motion vector (MV) yielding the least distortion. However, this often will not give the best image quality for a given bit rate, because it may select long motion vectors that need many bits to transmit. It also does not help determining how subdivision should be performed, because the smallest blocks will always minimize the distortion, even if the multiple MVs may use larger amount of bits and increase the bit rate. For this reason, H.264/AVC uses the cost function J, rather than SAD, as the measure of prediction error in selecting the best matching block [19] [41] [42]. The RD cost function J is defined as

$$J(mv, \lambda) = SAD(s, c(mv)) + \lambda R(mv - pmv) \tag{2.1}$$

where $mv = (mv_x, mv_y)^T$ is the current MV, $pmv = (pmv_x, pmv_y)^T$ is the predicted MV, and $SAD(s, c(mv))$ is the sum of absolute differences between current block s and candidate block c for a given motion vector mv , λ is the Lagrangian multiplier which is a function of quantization parameter (QP) and R(mv-pmv) is the number of bits to code the MVD. In H.264, the Lagrange multiplier for motion estimation is empirically calculated by the following formula [43]:

$$\lambda = 0.92 \times 2^{(QP-12)/6} \tag{2.2}$$

2.8 Rate-Distortion Optimized (RDO) Mode Decision

Since H.264/AVC includes many more coding modes than that in earlier standards, it is even more important that the mode decision be correct otherwise with sub-optimal choices, some of the coding efficiency benefits of H.264/AVC may be lost. However, as we will see, thus far the additional gains can be extracted only at the expanse of considerable increase in encoding complexity. To take the full advantages of all modes, the H.264/AVC encoder can determine the mode that meets the best RD tradeoff using RD optimization mode decision scheme. Then the best mode for every block that produces the minimum rate-distortion cost is given by

$$J_{RD}(S,C,m|QP) = SSD(S,C,m|QP) + \lambda_m R(S,C,m|QP) \tag{2.3}$$

where QP is the macroblock (MB) quantization parameter, λ_m is the Lagrangian multiplier and m is the candidate mode. A strong connection between the local Lagrangian multiplier and the QP was found experimentally as [10] [27]

$$\lambda = 0.85 \times 2^{(QP-12)/3} \tag{2.4}$$

In (2.2), SSD is the sum of squared difference between the original blocks S and the reconstructed block C, and it is expressed by

$$SSD = \sum_{i=1}^{4} \sum_{j=1}^{4} (S_{ij} - C_{ij}) \tag{2.5}$$

Where s_{ij} and c_{ij} are the (i, j)th elements of the current original block S and the predicted block C. The R in (2.2) can be written as

$$R = R_{header} + R_{motion} + R_{res} \tag{2.6}$$

where R_{header} , R_{motion} and R_{res} means the number of bits need to encode the header information, motion vectors and quantized residual block, respectively. In (2.2), mode (m) is chosen from the set of potential prediction modes as below:

For Intra frame: $m \in$ (Intra 4×4, Intra16×16)

For Inter frame: $m \in$ (SKIP, Inter 16×16, Inter 16×8, Inter 8×16, Inter P8×8, Intra 4×4, Intra 16×16)

SKIP mode means that no motion or residual bits will be encoded (only the mode indicator is transmitted) [36]. Inter P8×8 mode actually contains four sub-modes. A similar rate-distortion cost function is employed to determine the optimal sub-mode that achieves minimal rate-distortion cost. Once the best sub-mode is found, the minimal cost for the MB is evaluated by searching all the possible modes.

Inter P8 $\times$ 8 $\in$ (Inter 8 $\times$ 8, Inter 8 $\times$ 4, Inter 4 $\times$ 8, Inter 4 $\times$ 4)

It is found that the rate-distortion function JRD costs a lot of computation in real encoding as it requires the following computations:

1. Compute the predicted block: P
2. Compute the residual block: $Res = S\breve{}P$
3. Integer and scale transform of the residual block $F = IT(Res)$
4. Quantize the transformed residual block: $F^* = Q(F)$
5. Entropy coding of the quantized and transformed residual block to determine the bit-rate for encoding the block: $R = EC(F^*)$
6. Inverse quantize the quantized and transformed residual block: $F' = Q^{-1}(F^*)$
7. Inverse integer and scale transform the dequantized block: $R'es = IT^{-1}(F')$
8. Compute the reconstructed image block: $C = R'es + P$
9. Calculate SSD between original block Sand reconstructed block C
10. Calculate the cost function : $J_{RD} = SSD + \lambda.R$

The H.264 encoder computes this rate-distortion optimization process for every macroblock with all possible modes as shown in Fig.2.12. All of these processing explains the high computational complexity of $J_{RD}cost$ calculation. Hence, the cost function will make H.264/AVC impossible to be realized in real-time applications. To reduce the computation for these advanced coding technique become the major research task in video coding now.

2.9 Integer Transform, Quantization and Entropy Coding

The practical implementation of the DCT and quantization process in H.264/AVC is a little bit different from (6.1) and its architecture is shown in Fig.2.13. The DCT is implemented by ICT with scaling factors for complexity reduction which can be expressed as

$$F = DCT(D) = C_f D_f^T \bigotimes Q_{forw} = ICT(D) \bigotimes Q_{forw} = F^* \bigotimes Q_{forw} \tag{2.7}$$

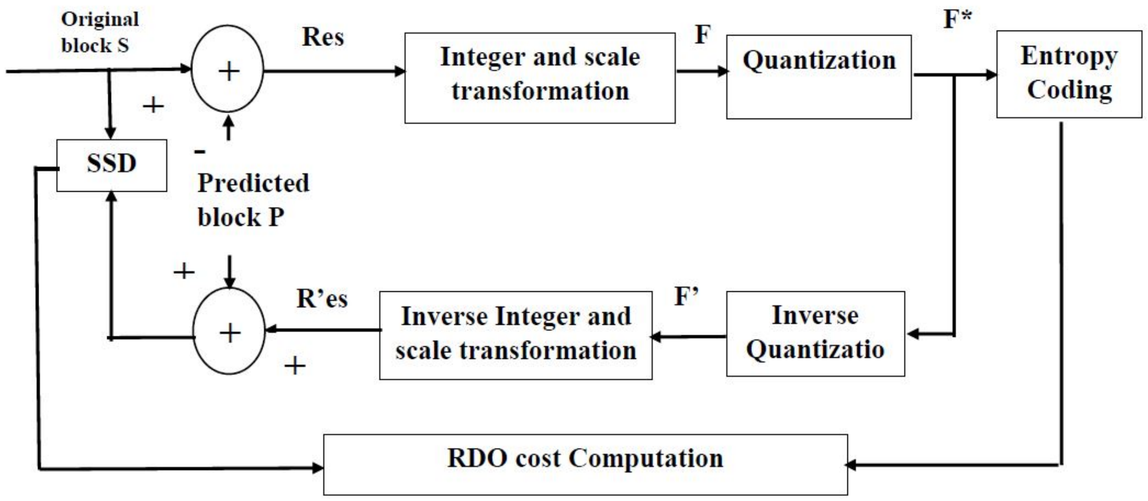


Fig. 2.12 Block diagram Rate-distortion cost computation.

where, C_f is called ICT core matrices, Qforw is called scaling factors and F^* is the ICT transformed block. The symbol indicates the operator that each element of (or F^*) is multiplied by the scaling factor in the corresponding position. The forward core and scale transform matrices are defined as

$$C_f = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}$$

$$Q_{forw} = \begin{bmatrix} a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \\ a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \end{bmatrix}$$

where $a = 1/2, b = \sqrt{2/5}$. The purpose is to reduce the computation complexity because the core transform can be realized by shift and addition operations only without multiplication. The quantization process of $Z = Q(F)$ for the transformed residual block F can be expressed as rounding operation on each coefficient of F:

$$z_{ij} = round(f_{ij}/\Delta) \tag{2.8}$$

where z_{ij} and f_{ij} are coefficients of the quantized transform and unquantized transform blocks of Z and F, respectively. Δ is the quantization step size, which is determined by the QP factor. On the other hand, the inverse quantization process of $\hat{F} = Q^{-1}(Z)$ can be expressed as scaling operation on each coefficients of Z:

$$\hat{F} = z_{ij}.\Delta \tag{2.9}$$

where $\hat{f}_{ij}$ are coefficients of the inverse quantized transformed block $\hat{F}$. In the inverse transform, the core matrix and scale matrix is not the same as those in forward transform.

$$\hat{D} = C_b^T(\hat{F} \bigotimes Q_{back})C_b \tag{2.10}$$

where C_b and Q_{back} are defined as

$$C_b = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1/2 & -1/2 & -1 \\ 1 & -1 & -1 & 1 \\ 1/2 & -1 & 1 & -1/2 \end{bmatrix}$$

$$Q_{back} = \begin{bmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{bmatrix}$$

In the H.264/AVC, scale transform and quantization are combined together to further reduce computational complexity.

$$z_{ij} = Q(f_{ij}^*) = round(f_{ij}^*.q_{ij}/\Delta) \tag{2.11}$$

where, q_{ij} are the scale coefficients of the Q_{forw} matrix and f_{ij}^* are coefficients of the ICT transformed block F^* using the ICT core matrix.

The H.264/AVC specifies the mathematical formulae of the quantization process. The quantization is also called scaling in the standard. The scale factor for each element in the sub-block varies as a function of the quantization parameter associated with the macroblock (MB) that contains the sub-block [43], and as a function of the position of the element within the sub-block. In H.264/AVC, two methods of entropy coding are supported [44]. The simpler entropy coding method uses a single infinite-extent code word table for all syntax elements except the quantized transform coefficients. Thus, instead of designing a different VLC table for each syntax element, only the mapping to the single code word table is customized according to the data statistics. The single code word table chosen is an exp-Golomb code with very simple and regular decoding properties.

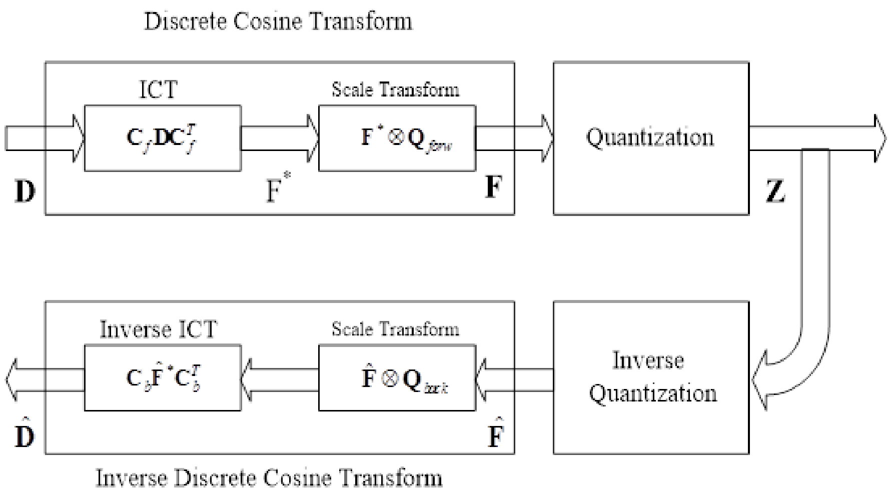


Fig. 2.13 Architecture of ICT and Quantization.

For transmitting the quantized transform coefficients a more efficient method called Context-Adaptive Variable Length Coding (CAVLC) [45] is employed. In this scheme, VLC tables for various syntax elements are switched depending on already transmitted syntax elements. Since the VLC tables are designed to match the corresponding conditioned statistics, the entropy coding performance is improved in comparison to schemes using a single VLC table.

The efficiency of entropy coding can be improved further if the Context-Adaptive Binary Arithmetic Coding (CABAC) is used [46]. Encoding with CABAC consists of three stages—binarization, context modeling and adaptive binary arithmetic coding. Fig. 2.14 shows a high level block diagram of CABAC encoder showing these various stages and their interdependence.

CABAC uses four basic types of tree structured codes tables for binarization. Since these tables are rule based, they do not need to be stored. The four basic types are the unary code, the truncated unary code, the kth order exp-golomb code, and, the fixed-length code. CABAC also uses four basic types of context models based on conditional probability.

The first type uses a context template that includes up to two past neighbors to the syntax element currently being encoded. For instance modeling may use a neighbor immediately before and an immediately above the current element, and further, the modeling function

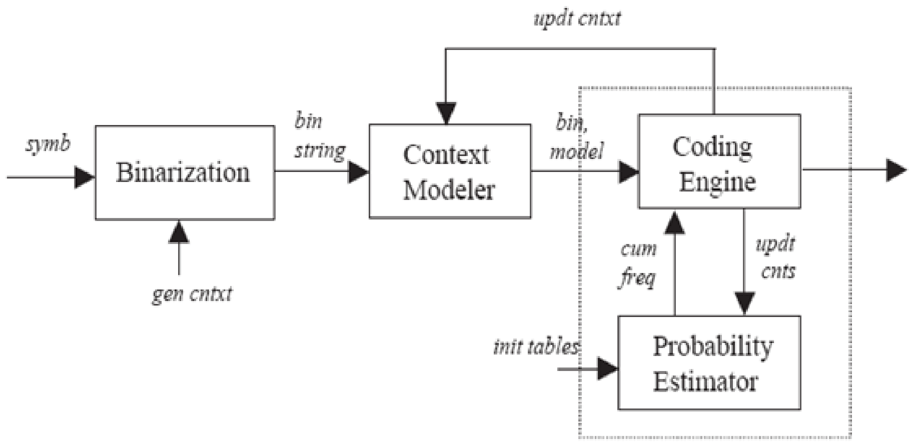


Fig. 2.14 CABAC encoder block diagram.

may be based on bin-wise comparison with neighbors. For instance modeling may use a neighbor immediately before and an immediately above the current element, and further, the modeling function may be based on bin-wise comparison with neighbors. The second type of context model is used only for syntax elements of MB type and sub-MB type and uses prior coded ith bins for coding of ith bin. The third and fourth types of context models are used for residual data only and are used for context categories of different block types. The third type does not rely on past coded data but on the position in the scanning path, and the fourth type depends on accumulated levels.

The arithmetic coding engine used is similar to that used by other arithmetic coders and generally involves, based on the probability of a symbol to be coded to perform recursive subdivision to fractional accuracy of an existing interval that initially spans the range from 0 to 1. Since the multiplication operation used in interval subdivision is a reason for increased complexity of arithmetic coders, a multiplication free state transition table based approach is used by CABAC.

2.10 De-blocking Filter

One particular characteristic of block-based coding is the accidental production of visible block structures. Block edges are typically reconstructed with less accuracy than interior pixels and “blocking” is generally considered to be one of the most visible artifacts with the present compression methods. For this reason, H.264/AVC defines an adaptive in-loop deblocking filter [47], where the strength of filtering is controlled by the values of several syntax elements. Adaptive de-blocking filter operates on the horizontal and vertical block edges within the prediction loop in order to remove artifacts caused by block prediction errors. The filtering is generally based on 4×4 block boundaries, in which two pixels on either side of the boundary may be updated using a different filter. The rules for applying the de-blocking filter are intricate and quite complex, however, its use is optional for each slice (loosely defined as an integer number of macroblocks). Nonetheless, the improvement in subjective quality often more than justifies the increase in complexity.

Table 2.4 shows the results of experiments with and without loop filter in coding of CIF test set with JM. Data is collected from reference [21]. The results show that the case of "no loop filter" as compared to having "default loop filter" performs very close, mainly produces bit-rate higher by up to 1%, but at the most 2% higher in bit-rate. So, at least, the bit-rate savings from loop filter is not significant. Since the main purpose of loop filter is in blockiness reduction, as expected, its benefit is in improvement of visual quality, especially when bit-rates are lower.

2.11 Hybrid Video Coding Scheme

In contrast with older MPEG-1/2/4 standards, the H.264 deblocking filter is not an optional additional feature in the decoder. It is a feature on both the decoding path and on the encoding path, so that the in-loop effects of the filter are taken into account in reference macroblocks used for prediction. When a stream is encoded, the filter strength can be selected, or the filter can be switched off entirely. Otherwise, the filter strength is determined by coding modes of adjacent blocks, quantization step size, and the steepness of the luminance gradient between blocks. The filter operates on the edges of each 4×4 or 8×8 transform block in the luma and chroma planes of each picture.

Hybrid video coding is the basis of all the video coding standards. The term 'hybrid' denotes a mix of intra-frame and inter-frame coding. A generic hybrid video encoder is shown in Fig.2.15. For all the standards, they commonly consist of some basic functional elements like the transformer, quantizer and variable length encoder. A raw video frame Fn is input to the encoder. Each frame is then partitioned into a number of blocks with a size of 16×16 pixels that they are called macroblocks (MB). The encoder processes the MBs one by one. A MB is encoded either in intra or inter mode. For both modes, a prediction P will be produced. In inter mode, P_i is formed by compensating the motion corresponding to the reference frames, Fn-1, Fn-2, Fn-3,........, stored in frame memory. The motion estimation [22],[37] finds the best match of the current block from one or more reference frames and

Table 2.4 Comparison of De-blocking filter and non De-blocking filter [21]

Sequence	Loop filter 5 ref	No loop filter , 5 ref,		Loop filter, 3 ref		No loop filter, 3 ref	
	1 B-pic	1 B-pic		2 B-pic		2 B-pic	
	Bit-rate Y SNR	Bit-rate Y SNR		Bit-rate Y SNR		Bit-rate Y SNR	
	(kbps)(dB)	(kbps)(dB)		(kbps)(dB)		(kbps)(dB)	
Mobile	664.04 31.35	665.27	31.31	602.91	31.03	606.19	30.99
		(+0.19%)				(+0.54%)	
Tempete	524.88 32.40	528.30	32.39	488.70	32.11	490.11	32.07
		(+0.65%)				(0.29%)	
Table Tennis	445.56 34.38	452.02	34.41	422.82	34.19	427.54	34.23
		(+1.45%)				(+1.11%)	
Flower Garden	814.65 31.73	814.19	31.72	745.66	31.32	744.62	31.31
		(-0.06%)				(-0.14%)	
Bus	697.57 32.90	703.36	32.66	658.70	32.63	662.27	32.60
		(+0.83%)				(+0.54%)	
Football	710.99 34.57	722.57	34.53	709.66	34.33	719.60	34.28
		(+1.63%)				(+1.40%)	
Stefan	697.91 32.80	697.97	32.74	668.94	32.46	681.61	32.38
		(-0.08%)				(+1.89%)	

generates the motion vectors MV which represent the displacement of the block. In intra mode, P_i is predicted from the previous reconstructed neighboring MBs. The prediction is then subtracted from MB to produce the residual data D. The residual data is then discrete cosine transformed (DCT) and quantized for lossy compression to give X which contains the quantized DCT coefficients [40]. These coefficients and motion vectors are entropy encoded to form a bit stream with other side information like quantization factor and prediction mode. Finally, it is output for storage and transmission.

Motion estimation and motion compensation play very important roles in the hybrid coding scheme, since they can greatly reduce the temporal redundancy between adjacent video frames. Generally speaking, the adjacent pictures always share much similar MBs; thus, the current MB can be presented by the other one in the previous frame with very little

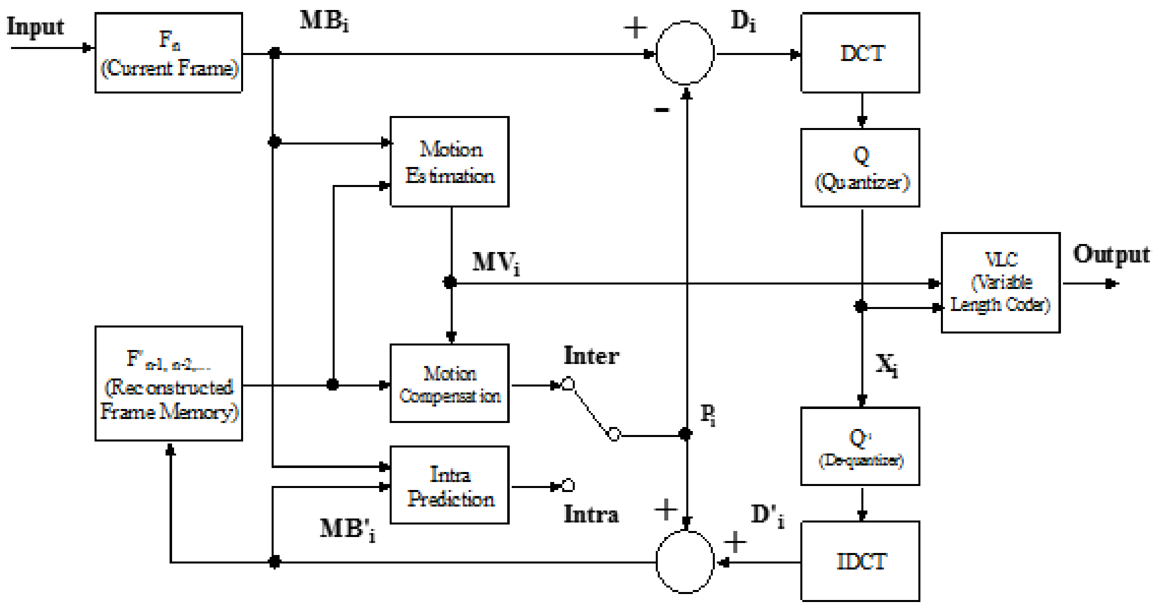


Fig. 2.15 Architecture of a basic hybrid video coding system.

difference. As illustrated in Fig.2.16, there are two adjacent frames from the video "reading". Though the two pictures are very similar, if the frame 1 is encoded with reference to frame 2, a large amount of residual data will be left without motion compensation, as shown in Fig.2.16(c). However, if the panning motion is compensated by finding the displacement with motion estimation, the remaining residual data will be very little, as shown in Fig. 2.16(d). For this reason, the necessary information to be coded is substantially reduced, and the huge video data can be effectively compressed.

A practical and widely-used method of motion estimation and compensation is called block-matching algorithm that has been universally adopted in almost all the video standard (MPEG series [17] [18] and H.26X [41]-[43]. It can compensate for movement of blocks of the current frame. The following procedure is carried out for each block of $M \times N$ samples in the current frame:

1. (Motion Estimation) Search an area in the reference frame to find a "best-matching" $M \times N$ block by comparing the $M \times N$ block in the current frame with all of the possible $M \times N$ blocks in the searching area. A popular matching criterion is the sum of absolute difference (SAD) between the current block and the candidate block. The candidate block that can minimize the SAD value is chosen as the best-matching block.

2. (Motion Compensation) The chosen candidate block becomes the predicted block of the current block and is subtracted from the current block to form a residual $M \times N$ block.

3. The residual block is encoded and transmitted; besides, the motion vector is also transmitted.

4. The decoder uses the received motion vector to form the predicted block and decodes the residual block. Then add the predicted block with the residual block to reconstruct the block.

2.12 Cost Function in H.264/AVC

In order to accelerate the coding process, the JVT reference software version JM 6.1d provides a fast SAD-based cost function [48]:

$$\begin{cases} J_{SAD} = SAD(S,P) + \lambda_1.4K \\ J_{SAD} = SAD(S,P) \end{cases} \quad \text{if intra } \; 4 \times 4 \text{ mode, } \; \text{otherwise} \tag{2.12}$$

where SAD(S,P) is sum of absolute differences between the original block S and the predicted block P. The λ_1 is also approximate exponential function of the quantization parameter (QP) which is almost the square of λ, and the K equal to 0 for the probable mode and 1 for the other modes. The SAD(S, P) is expressed by

$$SAD(S,P) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |s_{ij} - p_{ij}| \tag{2.13}$$

where s_{ij} and p_{ij} are the (i, j)th elements of the current original block S and the predicted block P, respectively. This SAD-based cost function could save a lot of computations as the distortion part is based on the differences between the original block and the predicted block instead of the reconstructed block. Thus, the processes of image block transformation, quantization, inverse quantization, inverse transformation and reconstruction can all be saved. In addition, the number of bits is estimated by constants either equal 4 or 0. Thus, the variable length coding using CAVLC or CABAC can also be saved. However, the expense of the computation reduction usually comes with quite significant degradation of coding

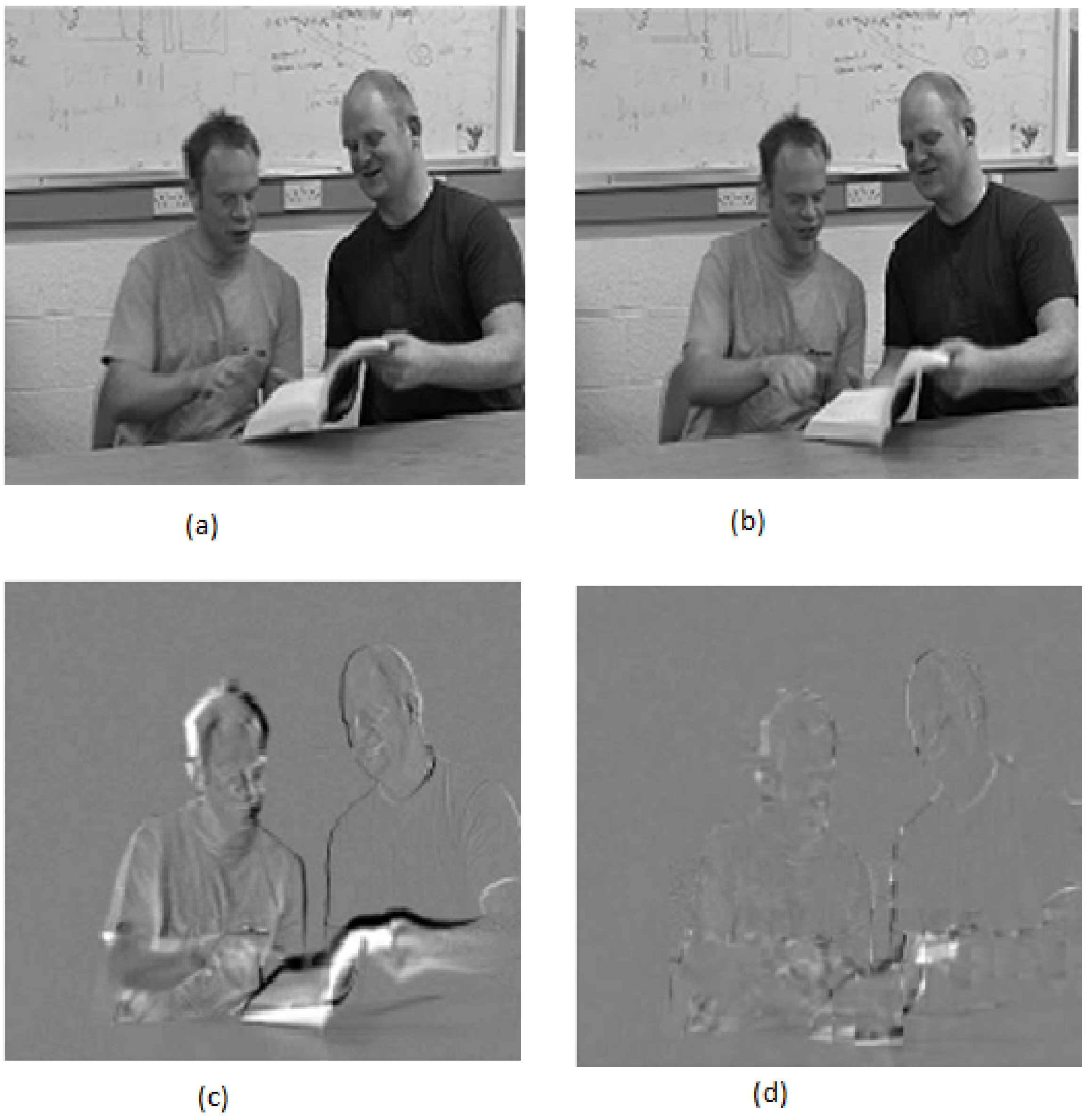


Fig. 2.16 (a) Frame 1, (b) Frame 2, (c) The residual picture by subtracting the frame 1 from frame 2 without motion compensation, (d) The residual picture by subtracting the frame 1 from frame 2 with motion compensation.

Table 2.5 H.264 Rate-distortion performance comparison using RDO-based, SAD-based and SATD-based cost functions in terms of PSNR (dB) and Rate (Kbps)

Sequence	QP	20			28			36		
		RDO	SAD	SATD	RDO	SAD	SATD	RDO	SAD	SATD
Akiyo	PSNR	44.44	44.19	44.27	38.78	38.55	38.68	33.16	32.93	33.10
	Rate	320.05	331.72	325.92	153.47	158.04	156.24	71.59	73.90	72.91
Foreman	PSNR	41.80	41.59	41.71	36.16	35.91	36.14	31.02	30.59	30.97
	Rate	683.60	701.15	693.59	251.97	259.69	261.85	100.47	106.98	106.71
Stefan	PSNR	41.86	41.56	41.62	35.01	34.67	34.79	28.56	28.09	28.33
	Rate	1309.08	1327.45	229.71	593.36	603.21	602.85	229.71	233.57	236.08

efficiency. To achieve better rate-distortion performance, JM6.1d also provided an alternative SATD-based cost function [48]:

$$\begin{cases} J_{SATD} = SAD(S,P) + \lambda_1.4K \\ J_{SATD} = SAD(S,P) \end{cases} \quad \text{if intra } 4\times4 \text{ mode, otherwise} \tag{2.14}$$

where SATD(S,P) is sum of absolute Hadamard-transformed [49] [50] difference between the original block S and the predicted block P, which is given by

$$SATD(S,P) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} |h_{ij}| \tag{2.15}$$

where h_{ij} are the (i, j)th element of the Hadamard transformed image block H which is the difference between the original block S and the predicted block P. The Hadamard transformed block H is defined as

$$H = T_H(S-P)T_H^T \tag{2.16}$$

With

$$T_H = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}$$

Experimental results show that the J_{SATD} could achieve better rate-distortion performance than the J_{SAD}, but it requires more computation due to the Hadamard transformation. If the fast Hadamard transform (FHT) is used, the computational requirement of the J_{SATD} can reduce half of the computation. However, the overall rate-distortion performance is still lower than the optimized J_{RD}. In addition, all these distortion expressions are based on the difference between original block and predicted block, so they avoid obtaining the reconstructed blocks. So, they cannot well predict the real distortion.

The rate-distortion performance comparison of H.264/AVC using rate-distortion- based, SAD-based and SATD-based cost functions for different QPs for 50 frames of three well-known test sequences in QCIF format in terms of PSNR and bit-rate is shown in Table 2.4. Compared with rate-distortion-based encoder, the performance degradation of both SAD-based and SATD-based cost functions are non-ignorable.

Chapter 3

MODIFIED INTRA PREDICTION SCHEME FOR H.264/AVC

In H.264/AVC intra coding, each 4×4 luma block is predicted as either one of the eight directional prediction modes or DC mode. DC mode is used to predict regions with no unified direction and the predicted pixel values are same and thus smooth varying regions are not well de-correlated. In order to address this issue, this chapter proposes a modified DC prediction (MDCP) mode based on the distance between the predicted and reference pixels. On the other hand, using the nine prediction modes in intra 4×4 block unit can reduce spatial redundancies, but it may needs a lot of overhead bits to represent the prediction mode of each 4×4 block. In order to reduce the number of overhead bits and computational cost, a modified intra 4×4 prediction mode scheme for H.264/AVC is proposed in this chapter. This can improve the prediction accuracy for areas without unified orientation. In this scheme, the upper right part of one 4×4 block mainly employs vertical prediction while the lower left part mainly uses horizontal prediction, predicting both vertical and horizontal directions in one block. The proposed scheme uses simple prediction techniques with fixed weighting coefficients. Experimental results show that MDCP is very competitive while comparing to others Intra 4×4 prediction algorithms. The proposed method not only improves the Rate-Distortion (RD) performance but also reduce the computational complexity of H.264/AVC intra encoder.

3.1 Literature Review and Challenges

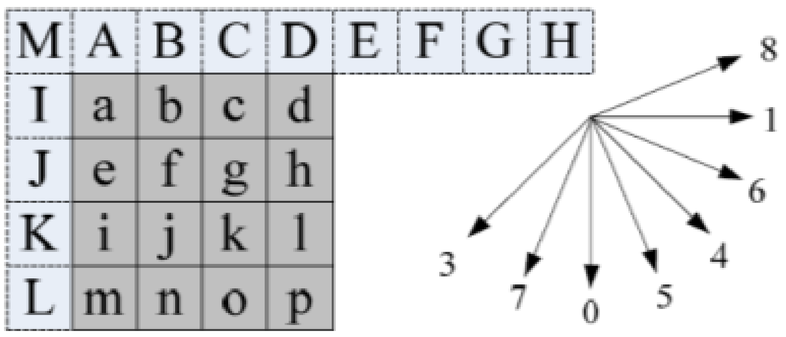
Joint Video Team (JVT) of ITU-T Video Coding Experts Group (VCEG) and ISO Motion Picture Experts Group (MPEG) has released the latest standard for video coding, which is known as H.264 or MPEG-4 Part 10 Advanced Video Coding (AVC) [51]. The directional spatial prediction for intra coding is one of the significant contributions of H.264/AVC. In contrast with previous standards, intra prediction in H.264 is applied in spatial domain instead of transform domain, which utilizes the correlation between adjacent blocks to remove spatial redundant information in an image.

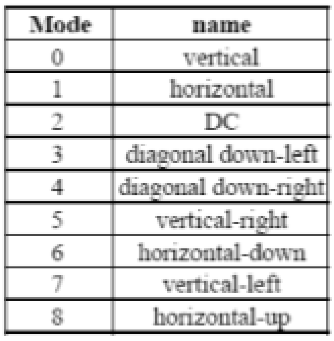
Many efforts have been done to simplify the intra coding in H.264. Fast intra mode decision algorithms were proposed to reduce the number of modes that needed calculation according to some criterion [52] [53]. Besides, fewer efforts on the intra prediction modes innovations were raised, either by altering the reference pixels [54] or by exchanging the order of intra prediction [55]. Most of the algorithms can reduce the time complexity remarkably, but few can improve the coding efficiency. Recently, distance-based weighted prediction (DWP) method and its simplified version integral DWP (iDWP) [56] were proposed to improve prediction efficiency, where the weighted prediction is done by coefficients which are inversely proportional to the distances between current pixel and its reference pixels. Although these methods give some improvement, they are computationally complex. This inspires us to propose a novel intra prediction mode called weighted cross prediction (WCP), which combines the vertical and horizontal predictions to replace the DC mode in Intra 4×4 prediction. The fixed weighting coefficients in our approach have a lower computation complexity with a comparable performance to DWP and iDWP. This inspires to develop an improved intra prediction which combines the vertical and horizontal predictions to replace the DC mode in Intra 4×4 prediction. The fixed weighting coefficients in our approach have a lower computation complexity with a comparable performance to DWP and iDWP.

In this chapter, a modified DC prediction mode called MDCP is proposed to reduce the number of residual bits. This proposed method can reduce the number of intra mode bits and computational complexity of encoder.

3.2 Intra Prediction Technique in H.264/AVC

The intra predictions in H.264/AVC are performed in a block-based manner, by referring to the neighboring samples of previously coded blocks which are left to and/or above the block to be predicted. In luminance prediction, nine optional prediction modes are provided for every 4×4 block, named as vertical, horizontal, DC, diagonal-down-left, diagonal-down-right, vertical-right, horizontal-down, vertical-left and horizontal-up prediction. Vertical, horizontal and DC modes, which are specified similar to the modes in Intra 4×4, with





Mode	name
0	vertical
1	horizontal
2	DC
3	diagonal down-left
4	diagonal down-right
5	vertical-right
6	horizontal-down
7	vertical-left
8	horizontal-up

Fig. 3.1 Intra 4 × 4 prediction modes.

plane prediction compose all the four prediction modes for 16 × 16 luma block. The Intra 4 × 4 prediction is suitable for the parts with significant details, while the Intra 16 × 16 is applied to the smoother areas. The encoder chooses the best prediction mode to minimize the Lagrangian cost function, which takes both distortion and bit rate into consideration. Fig. 3.1 illustrates 16 samples of 4 × 4 block (labeled as a-p) which are predicted by previous-decoded samples in the neighboring blocks (labeled as A-M), when using the Intra 4 × 4 prediction. For vertical or horizontal modes, the pixel values are extrapolated by upper samples or left samples, while other directional modes utilize the linear weighted average of reference samples. For the DC mode, all the predicted pixels are formed by means of upper and left samples.

In Intra 4 × 4 prediction, each block can be predicted using either the DC mode or one of the eight directional modes. As we know, these eight directional modes are used to predict the regions with unified orientations. Therefore, it is better if the remainder, DC mode, can be used to predict some areas that the textures have no unified orientation. However, using DC mode to predict such areas is not very accurate because it uses one value to predict all pixels in the block, which can not show any kind of variations between them. Since the Intra 4 × 4 DC mode cannot provide accurate prediction for some areas without unified directions, we replace DC mode with MDCP mode, which is much simpler and efficient.

3.3 Proposed Modified DC Prediction for Intra 4 × 4 Block

In H.264/AVC, DC mode is used to predict regions with no unified direction and the predicted values of all pixels are same. But the correlation that exists between predicted pixels and reference pixels are not absolutely considered to predict the DC prediction mode. Thus, the prediction signal generated by DC prediction is not well matched to the original signal, and a large number of bits are required for encoding the difference between the predicted and original signal. It is well known that Gaussian-like distribution can approximate local intensity variations in smooth image region. Original DC prediction always considers the reference pixels are belong to upper and left reconstructed blocks. But the correlation between neighboring pixels would be attenuated while the distance is increased and negligible when pixels are far apart. Therefore, prediction accuracy is degraded if all of the pixels of a 4×4 block are predicted from upper and left blocks.

Statistics show that the vertical and horizontal predictions are more frequently used than other modes, implying higher correlations between the reference samples and the pixels to be predicted in these two directions [57]. Therefore, it is possible to enhance the intra prediction accuracy by employing more such directional predictions, which inspires to develop MDCP to replace DC mode in Intra 4×4, via combining vertical and horizontal predictions.

Furthermore, because only the upper and left reference samples are available, block is divided into three parts: the diagonal, upper right part and lower left part. Since the diagonal pixels have the equal distance between the upper and left reference, they can be predicted by both the corresponding upper and left reference samples. However, the upper right pixels are closer to the upper reference while the lower left pixels closer to the left ones. Therefore, we mainly use the upper reference samples as the major component for the upper right part prediction while using the left reference samples for the lower left part. In other words, we use the vertical prediction as the major component for the upper right part, while employing the horizontal prediction as the major component for the lower left part.

Since correlation between pixels exists in both vertical and horizontal direction, we cannot only use the vertical or horizontal prediction directly in a 4×4 block. As a result, when we predict the upper right part, we use the left predicted pixel to adjust the vertical prediction, while using the upper predicted pixel as an adjuster to the horizontal prediction for the lower left part similar to [58].

As illustrated in Fig. 3.2, $P_{i,j}$ $(0 \leq i, 0 \leq j)$ denotes the predicted value of the pixel in the ith row and jth column of current 4×4 block. U_j and L_i denote the reference samples reconstructed from upper and left blocks respectively. In order to make a weighted prediction, a larger coefficient $3/4$ is assigned to the major reference samples while a smaller coefficient $1/4$ to the adjustment ones.

The denominators 4 make the division operation to be implemented by shift operation. $P_{i,j}$ can be calculated as:

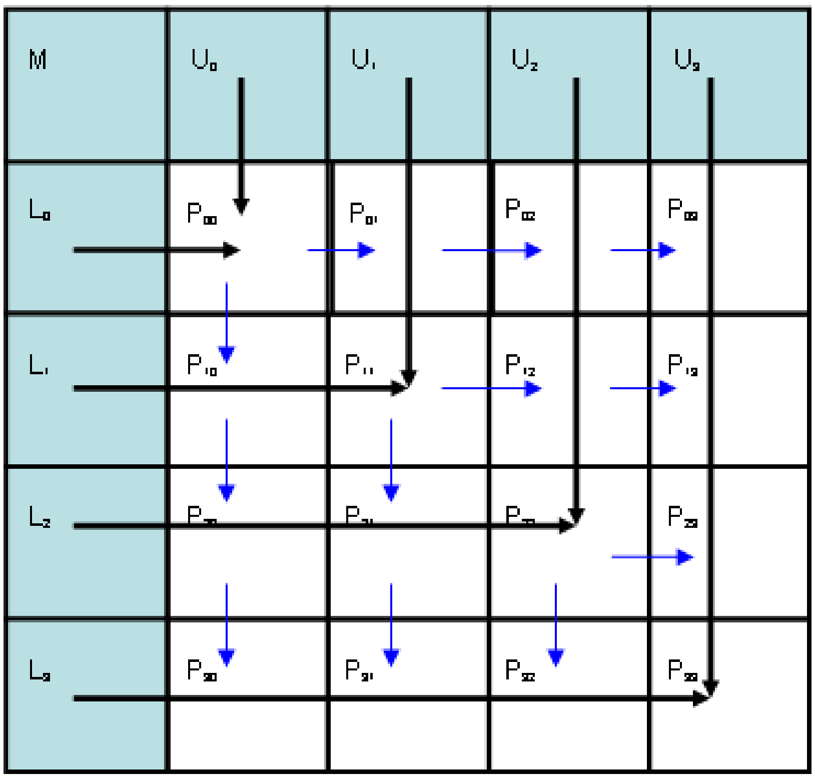


Fig. 3.2 Illustration of Intra 4 × 4 prediction.

$$P_{(i,j)} = \begin{cases} 3(U_j + P_{i,j-1} + 2) >> 2 = [(U_j << 1) + U_j + P_{i,j-1} + 2] >> 2 & \text{if } i < j; \\ (L_i + U_j + 1) >> 1 & \text{if } i = j; \\ 3(L_j + P_{i,j-1} + 2) >> 2 = [(L_j << 1) + L_j + P_{i,j-1} + 2] >> 2 & \text{if } i > j. \end{cases} \quad (3.1)$$

Since some of the predicted values are based on the other predicted values, we have to compute the predicted values in the following order. The diagonal pixels should be computed first, which are only based on U_j and L_i. Then, the upper right and lower left parts can be computed. The modified expressions of MDCP are listed as follows. It can be seen that the longer the distance between the reference sample and the pixel is, the smaller the coefficient becomes.

$$P_{0,0} = \frac{1}{2}U_0 + \frac{1}{2}L_0 \quad (3.2)$$

$$P_{1,1} = \frac{1}{2}U_1 + \frac{1}{2}L_1 \quad (3.3)$$

$$P_{2,2} = \frac{1}{2}U_2 + \frac{1}{2}L_2 \tag{3.4}$$

$$P_{3,3} = \frac{1}{2}U_3 + \frac{1}{2}L_3 \tag{3.5}$$

$$P_{0,1} = \frac{3}{4}U_1 + \frac{1}{4}P_{0,0} = \frac{3}{4}U_1 + \frac{1}{8}(U_0 + L_0) \tag{3.6}$$

$$P_{0,2} = \frac{3}{4}U_2 + \frac{1}{4}P_{0,1} = \frac{3}{4}U_2 + (\frac{3}{16}U_1 + \frac{1}{32}U_0 + \frac{1}{32}L_0) \tag{3.7}$$

$$P_{1,2} = \frac{3}{4}U_2 + \frac{1}{4}P_{1,1} = \frac{3}{4}U_2 + \frac{1}{8}(U_1 + L_1) \tag{3.8}$$

$$P_{0,3} = \frac{3}{4}U_3 + \frac{1}{4}P_{0,2} = \frac{3}{4}U_3 + (\frac{3}{16}U_2 + \frac{3}{64}U_1 + \frac{1}{128}U_0 + \frac{1}{128}L_0) \tag{3.9}$$

$$P_{1,3} = \frac{3}{4}U_3 + 1/4P_{1,2} = \frac{3}{4}U_3 + (\frac{3}{16}U_2 + \frac{1}{32}U_1 + \frac{3}{4}L_1) \tag{3.10}$$

$$P_{2,3} = \frac{3}{4}U_3 + \frac{1}{4}P_{2,2} = \frac{3}{4}U_3 + \frac{1}{8}(U_2 + L_2) \tag{3.11}$$

$$P_{1,0} = \frac{3}{4}L_1 + \frac{1}{4}P_{0,0} = \frac{3}{4}L_1 + \frac{1}{8}(U_0 + L_0) \tag{3.12}$$

$$P_{2,0} = \frac{3}{4}L_2 + \frac{1}{4}P_{1,0} = \frac{3}{4}L_2 + (\frac{3}{16}L_1 + \frac{1}{32}U_0 + \frac{1}{32}L_0) \tag{3.13}$$

$$P_{2,1} = \frac{3}{4}L_2 + \frac{1}{4}P_{1,1} = \frac{3}{4}L_2 + \frac{1}{8}(U_1 + L_1) \tag{3.14}$$

$$P_{3,0} = \frac{3}{4}L_3 + \frac{1}{4}P_{2,0} = \frac{3}{4}L_3 + (\frac{3}{16}L_2 + \frac{3}{64}L_1 + \frac{1}{128}U_0 + \frac{1}{128}L_0) \tag{3.15}$$

$$P_{3,1} = \frac{3}{4}L_3 + \frac{1}{4}P_{2,1} = \frac{3}{4}L_3 + (\frac{3}{16}L_2 + \frac{1}{32}U_1 + \frac{1}{32}L_1) \tag{3.16}$$

$$P_{3,2} = \frac{3}{4}L_3 + \frac{1}{4}P_{2,2} = \frac{3}{4}L_3 + \frac{1}{8}(U_2 + L_2) \tag{3.17}$$

We also compare the computation complexity between iDWP and MDCP. The computation complexity of iDWP to predict one pixel includes 1 multiplication, 2 additions, 1 subtraction, and 2 shifts, which is an approximation to DWP [53] The number of operation to predict one pixel in MDCP is 2.75 additions and 1.75 shifts in average, which is less complex than iDWP.

3.4 Experimental Setup and Results

The performance of the proposed method is evaluated by using JM 10.2 [59] reference software. Different types of video sequences with different resolutions are used as test materials. A group of experiments were carried out on the test sequences with different quantization parameters (QPs). The performance of MDCP is compared with intra prediction scheme of H.264 in terms of the average PSNR (dB) and the bit rate (kbits/s) for several test sequences including both CIF and QCIF format. In the experiment, all the sequences have 100 frames (30Hz), compressed with all Intra prediction by applying four quantization parameters: QP = 20, 24, 28 and 32. The test conditions are indicated as follows:

1. Intra 16×16 and Intra 4×4 are used;
2. RD Optimization is used;
3. The entropy coding method is CABAC;
4. The 8×8 transform is disabled;
5. The adaptive rounding is disabled.

Since the proposed method is for low complexity intra prediction, we compare the result of MDCP with iDWP here. The simulation results for MDCP and iDWP are listed in Table 3.1 and Table 3.2. The increase of PSNR is calculated when the bit rates are equal while the decrease of bit rates is calculated when the PSNRs are the same. The rate-distortion (RD) curves of Bus and Salesman sequences are shown in Fig. 3.3 and Fig. 3.4.

A constant performance gain is obtained by MDCP over H.264 with an average PSNR improvement of 0.06 dB which reaches its maximum of 0.09 dB in CIF sequences. In QCIF sequences, the average PSNR increase is 0.06 dB. It can be seen that the result of MDCP is comparable to iDWP. In mobile sequence, the performance is even better than iDWP. In addition, MDCP shows a better performance in CIF than in QCIF sequences suggesting its better suitability for high resolution sequences.

Table 3.1 Simulation results of CIF sequences

	MDCP		iDWP	
Sequence (CIF)	△ PSNR (dB)	△ bitrate (%)	△ PSNR (dB)	△ bitrate (%)
Bus	0.09	-0.96	0.13	-1.35
Stefan	0.08	-0.78	0.10	-0.95
Mobile	0.06	-0.48	0.06	-0.44
Foreman	0.02	-0.23	0.02	-0.30
Average	0.06	-0.61	0.08	-0.76

Table 3.2 Simulation results of QCIF sequences

	MDCP		iDWP	
Sequence (QCIF)	△ PSNR (dB)	△ bitrate (%)	△ PSNR (dB)	△ bitrate (%)
Salesman	0.07	-0.86	0.11	-1.28
News	0.06	-0.55	0.09	-0.89
Claire	0.04	-0.53	0.06	-0.67
Average	0.06	-0.65	0.09	-0.95

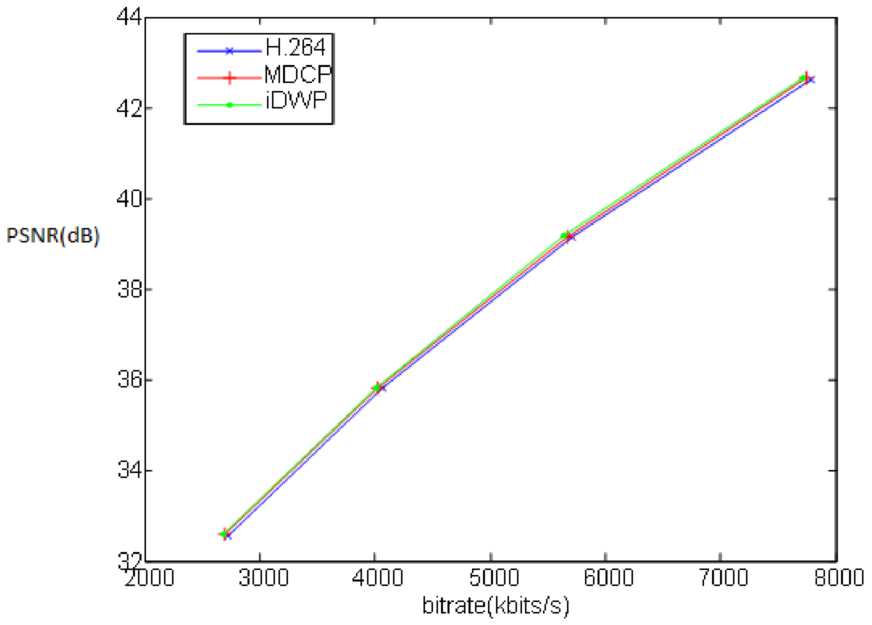


Fig. 3.3 RD curve for "Bus" sequence.

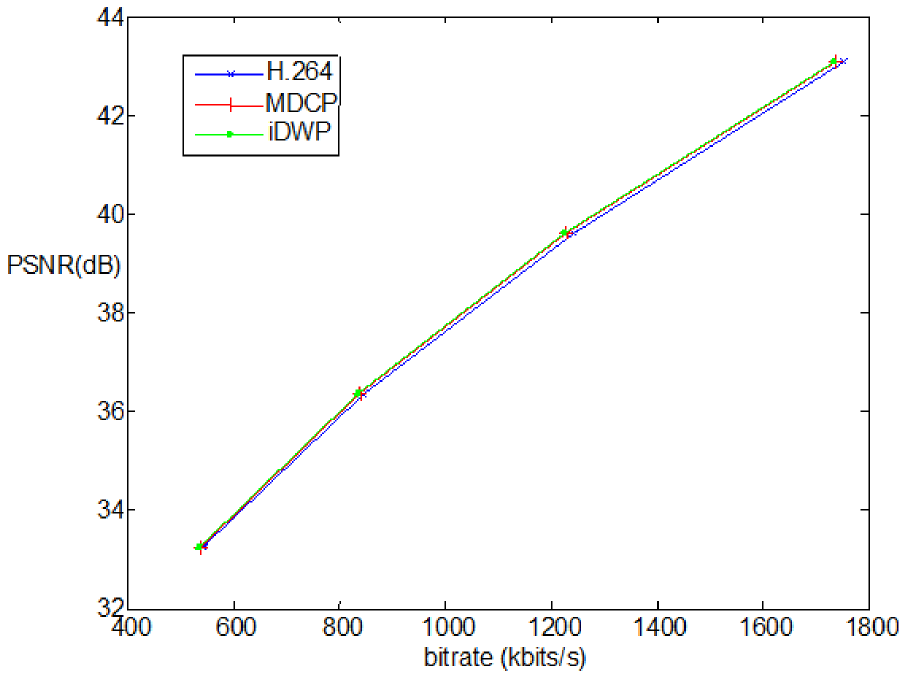


Fig. 3.4 RD curve for “Salesman” sequence.

3.5 Summary

In this chapter, a modified DC prediction (MDCP) is proposed to replace DC mode in Intra 4×4 to improve the RD performance of H.264/AVC intra encoder to solve the problem that DC mode used in the regions with no unified direction. To improve the prediction accuracy, we utilize the vertical prediction as the major approach on the upper right part, while the horizontal prediction as the major approach on the lower left part, which includes both vertical and horizontal predictions in one block. The experimental results show improvements compared to the original H.264 with DC mode and MDCP gives comparable RD performance to iDWP with lower computation complexity. The proposed methods not only improve the RD performance but also reduce computational complexity of H.264/AVC intra encoder.

Chapter 4

A MODIFIED Z-SCANNING SCHEME FOR AVS INTRA CODING

In traditional directional spatial prediction based video coding standards such as H.264/AVC, MPEG4 and AVS, zigzag scan is the fixed scan pattern which designed to organize quantized transform coefficients in order to bring the high-frequency components as more as possible, so that the coefficients can be encoded more efficiently using the entropy encoding. However, zigzag scan cannot efficiently organize the transform coefficients due to different residual energy distribution produced by different intra prediction. To resolve this problem, in this chapter, a modified scanning scheme named as Z-scan scheme is proposed to further improve intra coding efficiency for the AVS standard. In our method, traditional zigzag scan, horizontal z-scan, and vertical z-scan are used depending on the spatial prediction directions. It is relatively easy to implement our Z-scan scheme into AVS codec without changing the syntax. Experimental results demonstrate that the Z-scan scheme can remarkably reduce bit rates by approximately 2.1% compared with AVS codec using zigzag scan, while the PSNR of video sequences are maintained.

4.1 Literature Review

The Audio Video Coding Standard (AVS) is a China's national standard for media compression, which is developed by Audio and Video Coding Standard Working Group of China. Similar with H.264/AVC video coding standard, AVS is a block transform-based codec uses a run length coding technique to encode the quantized coefficients corresponding to a particular prediction residual block. Run length coding proceeds by zigzag scanning a block of quantized transform coefficients with a pre-defined pattern.

In AVS intra coding, there are five different intra prediction methods for 8×8 blocks are supported [60]. In general, different intra prediction method can introduce different prediction residual. Therefore, it is inefficient for coding the residual signal predicted from different directions with only zigzag scan being used.

Recently, to resolve this problem, many works have been done on developing a suitable san pattern for various video coding standards. For MPEG-4 intra coding, three block-level adaptive scans (alternate-vertical, alternate-horizontal and zigzag) for each intra blocks according to DC prediction direction was utilized in [61]. For H.264/AVC intra coding, [62]-[64] proposed to use two extra scan patterns for vertically predicted block and horizontally predicted block respectively, and the traditional zigzag scan was used for other prediction modes. Similarly, in [65], they proposed an adaptive scan method that used a different scan method for each intra prediction mode. For AVS standard, depending on the statistical distribution of prediction residual, an new scheme called MLASS that adaptively choose the best scan pattern for each macroblock to arrange the quantized DCT coefficients was presented in [66], moreover, it can not only be used for intra coding but also for inter coding.

In this chapter, we propose a new Z-scan scheme for AVS intra coding which can further improve the coding efficiency on bit rate saving. Within this scheme, according to the spatial prediction directions, two additional scan table called horizontal Z-scan and vertical Z-scan are used together with normal zigzag scan to organize coefficients.

4.2 Overview of AVS Intra Coding

AVS intra coding is similar with H.264/AVC except that the prediction process is conducted for each 8×8 luma/chorma block in the spatial domain. Residual spatial redundancy is removed by spatial prediction and transform coding. The general block diagram of the AVS intra encoder can be seen in Fig. 4.1.

4.2.1 Intra prediction and transform

For the intra prediction part in Fig. 4.1, intra blocks of intra frames and intra-coded macroblocks in P-frames and B-frames are spatially predicted from previously encoded and reconstructed neighbouring blocks. For the luma samples, there are five directional intra modes for each 8×8 block: Vertical (mode 0), Horizontal (mode 1), DC (mode 2), Down-left (mode 3) and Downright (mode 4). Fig. 4.2 illustrates the process of intra 8×8 prediction. To encode a luma block, 16 previously decoded pixels above and to the right of the current block together with 16 previously decoded pixels to the left and below of the current block will be used to predict current block pixels. The best prediction mode for each block in terms of RD performance is selected in mode selection part. After intra prediction, AVS performs a separable, 8×8 integer cosine transform (ICT) on the residual block for energy compaction and followed by a quantization to minimize coding complexity.

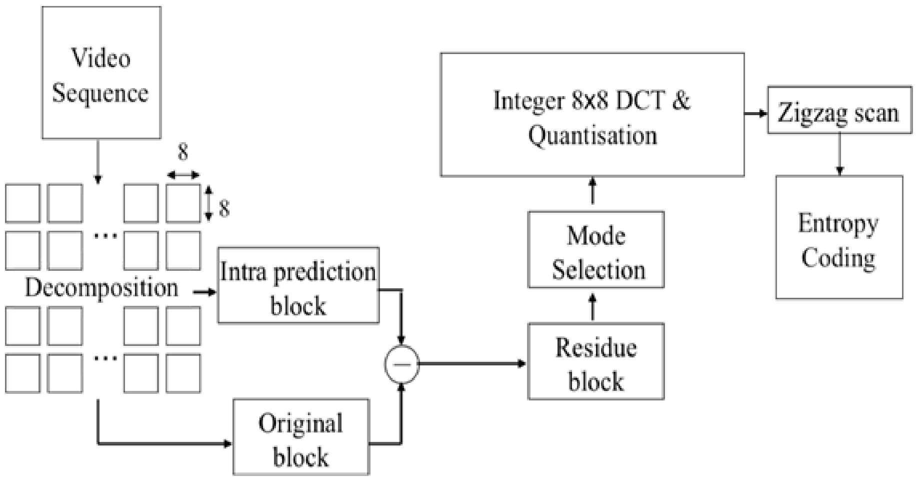


Fig. 4.1 Block diagram of AVS Intra encoder [65].

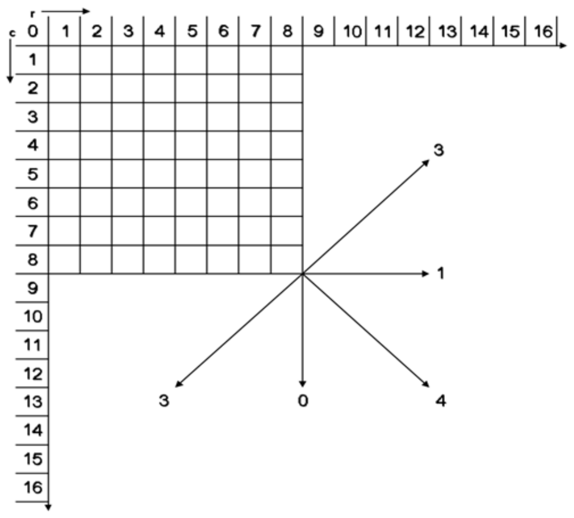


Fig. 4.2 Intra Prediction Modes for 8×8 blocks [66].

4.2.2 Zigzag Scan and Challenge

For binary coding of the quantized transform coefficients, these coefficients are mapped into a 1-D array following the zigzag order shown in Fig.3. Thus, bigger low-frequency coefficients are put in front of the high frequency coefficients that are usually zero-valued.

As we know, zigzag scan can efficiently group coefficients of transform typical image block for image coding systems, for instance, JPEG 2000. In directional spatial prediction based video coding systems, however, it may not be ideal for organizing transform coefficients. We have observed that the pattern of likely coefficients in a transform residual block shows some kind of local data dependency. Different texture regions in video sequence decides different optimal prediction mode, meanwhile, different intra prediction mode leads to different residual coefficient distribution. Consequently, one fixed scan pattern cannot be always ideal (in the sense of reducing the required bits of representing current block) for every block with various kinds of texture features. For example, blocks that use vertical or horizontal prediction modes as the best modes show different frequency change features from blocks that use the other prediction modes [62]. We, hereby, focus on the vertical prediction and horizontal prediction.

When a block is assigned the vertical prediction modes, the vertical predicted residual pixels will have a high correlation in the vertical direction; therefore, in the frontal up rows, the transform coefficients will have relatively large values which are not likely to be quantized to zero. In a similar way, when a block uses horizontal intra prediction, the transform coefficients in the frontal left columns are less likely to be quantized to zero due to their relatively large value.

However, zigzag scan used in AVS is utilized without considering the specific transform coefficients distribution characteristics that are stated above. As can be seen in Fig. 4.3, zigzag scan is particularly suitable for symmetric coefficient distribution, because it scans along the horizontal direction and the vertical direction with the same priority. Therefore, zigzag scan cannot efficiently organize coefficients while vertical prediction or horizontal predictions are being used. Meanwhile, the entropy coding efficiency will be decreased because less non-zero coefficients are grouped together. In order to further improve the coding efficiency, our target is to design a better scanning method that can take advantage of specific distribution features of different intra prediction modes. Based on above analysis, we proposed a new z-scan scheme. As shown in Fig.4.3, two scan patters are added to the proposed scheme: Horizontal Z-scan in Fig.4.4 and Vertical Z-scan in Fig.4.5.

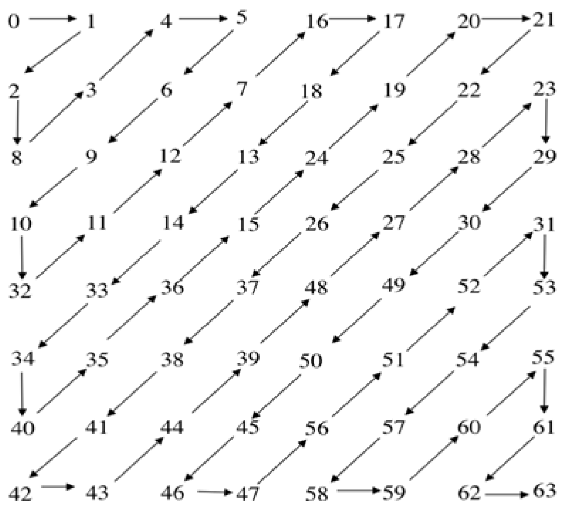


Fig. 4.3 Zigzag scan pattern [60].

4.3 Proposed Methodology of Z-Scan Scheme

In proposed method, when the vertical mode is selected to be the best prediction mode of a block, the Horizontal z-scan will be used to scan the transform coefficients. It starts from the left-top corner and scans the each coefficient as z-shape unit rows by rows from top to bottom, in this way, higher priority is given to non-zero coefficients in the horizontal direction. Besides, the Horizontal z-scan has the zigzag property that can take into account the symmetric increasing frequency while horizontally scanning.

Owing to this Horizontal scan, it is possible group more consecutive non-zero coefficients into the 1D scan sequence, meanwhile, it can group more consecutive zeros together which is helpful for the following entropy coding. Similarly, Vertical z-scan will be used when a block is intra predicted by horizontal mode. Also starting from the top-left position; it scans each position as z-shape unit columns by columns from left to right. All coefficients in the vertical direction are read faster than those in horizontal direction; non-zero coefficients can be grouped more efficiently in vertical direction.

When the current block uses the other three kinds of prediction modes: DC, Down-left and Down-right, the conventional zigzag scan will be used to organize the transform coefficients. According to the description above, the scan function of AVS standard is added into two new scan patterns, thus, we have three scan tables including zigzag scan exist in new AVS codec. The new scanning process can be performed as follows:

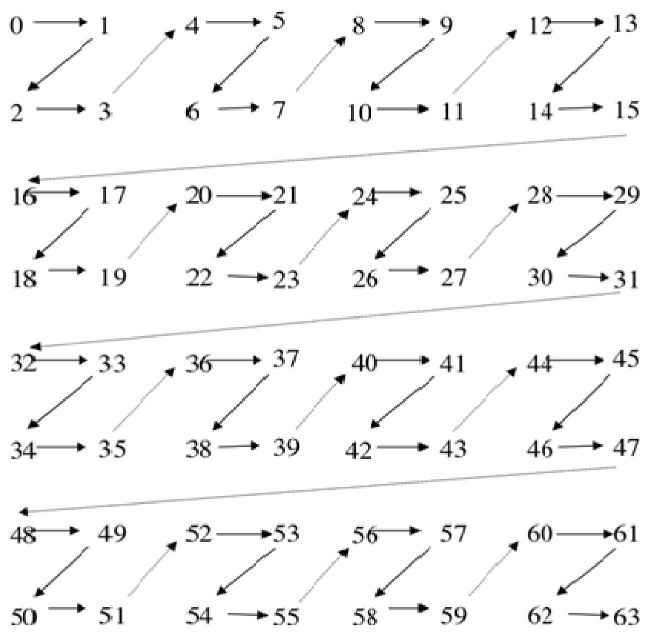


Fig. 4.4 8 × 8 Horizontal Z-scan.

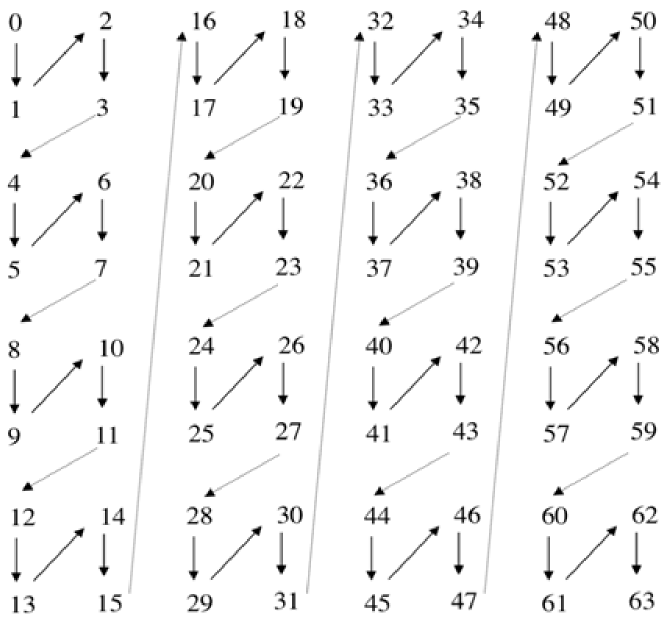


Fig. 4.5 8 × 8 vertical Z-scan.

Table 4.1 Test conditions for experimental setup

Test sequence	CIF: Foreman, Paris 4CIF: City, Soccer
Color format	YUV 4:2:0
Sequences parameters	Only intra coding; 200 frames; 30 fps
QP	28, 32, 36, 40
Hadamard transform	On
RD Optimization	On
Frame type	Progressive / Frame coding
Entropy coding	2D-VLC

- For each individual block, before scan, the prediction mode of the current block will be informed by the intra prediction function.
- Intra prediction mode will be used as the signal of deciding the scan patter. Blocks using Vertical prediction will choose Horizontal z-scan, blocks using Horizontal prediction will choose Vertical z-scan, and blocks using other three prediction modes choose zigzag scan.
- Scan the transform coefficients block with corresponding scan pattern.

It should be noticed that there are no extra bits for representing signal of intra prediction mode for current block, since it can be directly derived from intra prediction function just before scan function in AVS codec.

4.4 Experimental Setup and Results

For the experimental results the test model AVSsm 0.4 developed by China AVS group is used to evaluate the performance of proposed Z-scan scheme. All of test conditions can be found in Table 4.1.

Test results about four sequences with different QPs are illustrated in Table 4.2. The 2nd column shows the different QPs using in our experiments, it can be seen that our method outperforms conventional AVS for every QP, so that our scheme is quantization parameter independent. The last column demonstrates that, under every condition, the proposed z-scan scheme can averagely save approximately 2.07% bits compared to conventional AVS standard with only zigzag scan being used. Furthermore, it can achieve a maximum bit rates saving of 3.4%. More importantly, as shown in 3rd and 5th column, our z-scan scheme can obtain

Table 4.2 RD performance comparison of conventional Zigzag scan with proposed Z-scan method

Sequences	QP	PSNR Y (zigzag scan)	Bitrates (zigzag scan)	Proposed PSNR Y (Z-scan)	Proposed Bitrates (Z-scan)	Bit rates saving
Paris(CIF)	28	40.26	7026.98	40.23	6850.31	176.67, 2.6%
	32	37.70	5545.53	37.68	5391.78	153.75, 2.9%
	36	35.34	4378.29	35.32	4245.55	132.74, 3.1%
	40	32.85	3294.89	32.83	3186.08	108.81, 3.4%
Foreman(CIF)	28	41.04	3691.76	41.02	3635.25	56.51, 1.55%
	32	38.9	2676.11	38.89	2631.69	44.42, 1.69%
	36	37.00	1970.24	36.98	1937.46	32.78, 1.69%
	40	35.05	1399.91	35.03	1378.33	21.58, 1.57%
City(4CIF)	28	40.05	16384.10	40.02	16200.34	183.77, 1.13%
	32	37.77	12480.90	37.74	12312.00	168.92, 1.37%
	36	35.43	9197.84	35.41	9056.69	141.45, 1.56%
	40	33.37	6714.00	33.35	6599.26	114.74, 1.74%
Soccer(4CIF)	28	41.29	11788.30	41.27	11507.81	280.47, 2.44%
	32	39.17	8810.35	39.15	8611.50	198.85, 2.31%
	36	36.97	6344.38	36.95	6210.88	133.5, 2.15%
	40	35.02	4512.77	35.01	4425.26	87.51, 1.98%

similar PSNR values while reducing the bit rates significantly. For clear understanding the graphical representation of the percentage of gain of different video sequences at different QP values are given in Fig. 4.6.

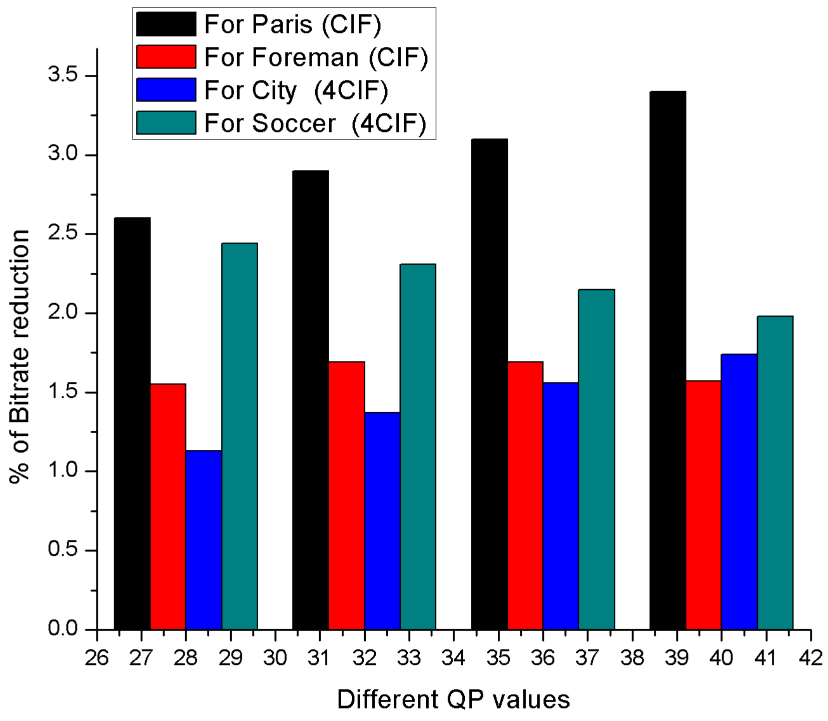


Fig. 4.6 Percentage of bit rate reduction for different video sequences at different QP values.

For each sequence, considering that the coarser quantization parameter being used the more coefficients will be quantized to zero. Therefore, it is easier for our z-scan method to efficiently group more zero coefficients together after less consecutive non-zero values. In this way, greater performance improvement can be obtained when the QP is higher. For City (4CIF) sequence, the video texture is so complex that it is less to utilize simple vertical or horizontal mode to predict. For this reason, there is not much improvement for City (4CIF) sequence.

For Foreman (CIF) sequence, there is even small room for coding efficiency improvement due to more flat areas and less complex image texture. For Paris (CIF) and Soccer (4CIF) sequences, they exhibit the common feature of relatively simple and fixed background. This feature leads to more blocks are predicted with vertical and horizontal modes. Thus, within the proposed method, more bit rates can be saved. Since improvement of intra coding is difficult to achieve, as widely known, the bit rates saving in our experiments is obviously significant especially almost without quality reduction.

4.5 Summary

In this chapter, a modified Z-scan scheme for directional spatial prediction of AVS intra coding is proposed. Each block chooses the appropriate scan mode from three scan tables according to its intra prediction mode. Experimental results demonstrated that after implementation of our new scan modes, the intra coding efficiency in terms of RD performance can outperform the conventional fixed zigzag scan pattern for all test sequences regardless of the test conditions. Besides, there is no extra computational cost in the proposed Z-scan implementation.

Chapter 5

RATE-BASED FAST MODE DECISION SCHEME IN H.264/AVC

In this chapter, a rate-based macroblock classification for fast mode decision in H.264 video coding standard is proposed. The main idea is to classify each macroblock into simple motion or complex motion contents based on the Inter 16×16 mode's residue block bit-rate and then according to the classification different mode searching orders with distinct early termination schemes are employed. This new algorithm is very simple for both hardware and software implementations without extra computational module. To speed up the intra mode decision, a new fast Intra 4×4 mode selection algorithm is also proposed by choosing most likely modes using the low complexity SATD cost as screening function. It is demonstrated by experimental results that the proposed rate-based fast algorithm can reduce 47% to 65% of the H.264 total encoding time with negligible degradation in the rate-distortion performance. While the rate-based algorithm combined with the fast intra mode selection method could further speedup 5% to 10% more of the encoding time with only little rate-distortion degradation.

5.1 Literature Review

H.264 standard achieves much higher compression performance than the present standards [17], such as MPEG-1, MPEG-2, MPEG-4 and H.261/H.263 [5-8]. In order to achieve the highest coding efficiency, H.264 employs rate distortion optimization (RDO) technique to get the optimal coding result in terms of maximizing coding quality and minimizing resulting data bits [7, 8]. Therefore, it has to encode the video sequence by covering all possible modes, including intra and inter prediction modes and allowing variable-size blocks in motion estimation, which contributes a lot to the prediction accuracy. The basic idea to bring in many prediction modes is that the blocks with higher motion details can be better encoded with smaller blocks; similarly, the blocks with less motion details can be encoded with larger block modes.

Although the variable block size technique can significantly improve the prediction accuracy, it also brings in great computation loading, since for each macroblock there are totally nine modes needed to be calculated in order to determine the best mode with lowest rate-distortion cost. To reduce the computation, several fast mode-decision algorithms have been developed and mainly focused on how to eliminate unnecessary modes. By using the Sobel operator, the edge direction map is successfully utilized for fast H.264 inter mode selection. It is based on local edge directional information to reduce the amount of calculations in intra prediction [67]. With the use of edge direction histogram derived from the edge map of the picture, only a small number of most possible intra prediction modes are selected for RDO calculation. In [68], amplitude of the edge vector is used to detect the homogeneous blocks whose textures in the region having similar spatial property. It is because homogeneous regions tend to move together and homogeneous blocks in the frame should have similar motion and should not be further split into smaller blocks. Thus, probable large blocks-size intra modes can be predicted to avoid blocks splitting for further computation. In [69], an objective measure is proposed between the intra and the inter modes to decide spatial and temporal correlation. If the current block has more correlation in spatial direction, the block has higher probability of being an intra block. Since the computational cost of intra mode decision is high, huge computational complexity can be reduced if we can accurately predict whether intra mode is best or not in advance. A motion history based algorithm [70] is proposed by using spatial motion activity to divide the location into 'active' regions and 'static' regions. Then the active regions are more likely to be encoded with smaller modes; while the static regions are inclined to larger modes. Recently, a motion content classification based fast mode decision algorithm was proposed in [71], which makes use of the spatial mode correlation to predict the most probable mode of current macroblock. The neighboring macroblocks are distributed specific weighing values according to their positions (generally the closer, the larger). Then combine their coding types together and predict the most probable mode of current block.

Most of the developed fast mode decision algorithms are involved additional computation such edge direction map, coefficients cost or mode predictor for eliminating less probable modes, which may increase the complexity in practical implementation especially using hardware. In this chapter, the bit-rate for encoding the Inter 16×16 mode is proposed for simple motion and complex motion blocks classification and different mode searching orders and termination schemes are employed to eliminate unnecessary modes. As the encoding of the Inter 16×16 mode is part of the rate-distortion optimization mode selection process, thus no additional operation module is required for the proposed fast algorithm.

5.2 Proposed Algorithm

5.2.1 Rate-based Macroblock Motion Content Classification

In H.264, most of the computation cost is spent on those modes which are not the final best modes; therefore, in order to reduce the computation, a good way is to exclude some unlikely modes in advance. Basically, the macroblock (MB) can be divided into two types: simple motion macroblock (SMB) and complex motion macroblock (CMB). This is a rough motion content classification since we cannot precisely determine the best mode in advance. If an MB is considered as an SMB, only large modes (SKIP, Inter 16×16, Inter 16×8, and Inter 8×16) are selected for the best mode. Similarly, a CMB only covers small block-size modes (Inter 8×8, Inter 8×4, Inter 4×8, Inter 4×4, Intra 4×4 and Intra 16×16). This mode search strategy could avoid a lot of computation especially when a MB is predicted as SMB. In such condition, the computation of the rate-distortion cost for small block-size modes can be saved which is more computationally intensive than large block-size modes. To take advantage of this mode search approach, still, we need to develop an efficient and accuracy measure for SMB and CMB classification. In H.264, rate-distortion cost is used to decide the best mode and it is defined as

$$J_{RD} = SSD + \lambda (R_{motion} + R_{residue}) \tag{5.1}$$

where λ is the Lagrange multiplier [72], SSD presents the sum of squared difference between original MB and reconstructed MB, R_{motion} represents the number of bits for encoding the motion vectors and head information, and $R_{residue}$ represents the number of bits for encoding the residue MB. Based on the definition of the cost function, we may consider using J_{RD}, SSD, R_{motion}, and $R_{residue}$ for the motion content classifications. As we want to have an efficient classifier, the total cost of J_{RD} and SSD are not good candidates. It is because the computation of the SSD is very computationally intensive which requires forward/inverse quantization, forward/inverse transformation and pixel-reconstructions processes. The computation of the H.264 rate-distortion cost function J_{RD} can be summarized as [73]:

1. Compute the predicted block: P

2. Compute the residual (difference) block: $D = S - P$

3. Discrete cosine transform the residual block

$$F = DCT(D) = T_{DCT} D T_{DCT}^T \tag{5.2}$$

Where

$$T_{DCT} = \begin{bmatrix} a & a & a & a \\ b & c & -c & -b \\ a & -a & -a & a \\ c & -b & b & -c \end{bmatrix}$$

$a = \frac{1}{2}, b = \sqrt{\frac{1}{2}}\cos(\frac{\Pi}{8}), c = \sqrt{\frac{1}{2}}\cos(\frac{\Pi}{8})$

Besides, an important property of T_{DCT} is that it is a unitary matrix. In H.264, discrete cosine transform (DCT) has been developed into integer cosine transform (ICT) in order to reduce the computation complexity.

$$F = DCT(D) = C_f D C_f^T \bigotimes Q_{forw} = ICT(D) \bigotimes Q_{forw} \tag{5.3}$$

where, C_f is called core matrices, Q_{forw} is called scaling factors. The symbol $\bigotimes$ indicates the operator that each element of $C_f D C_f^T$ is multiplied by the scaling factor in the corresponding position. The forward core and scale transform matrix are defined as

$$C_f = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}$$

With

$$T_H = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}$$

$$Q_{forw} = \begin{bmatrix} a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \\ a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \end{bmatrix}$$

where, $a = \frac{1}{2}, b = \sqrt{\frac{2}{5}}$. The purpose is to reduce the computation complexity because the core transform can be realized by shift and addition operation without multiplication. Then the operation $\otimes$ is combined in the process of quantization.

4. Entropy code of the Z to find the number of bit to encode the block: R = VLC(Z)

5. Inverse quantization: $F' = Q^{-1}(Z)$

6. Inverse transform the dequantized block: $D' = DCT^{-1}(F')$

7. Compute the reconstructed image block: $C = D' + P$

8. Calculate the cost function : $J_{RD} = SSD + \lambda.R$

Mathematically, the original block S and reconstructed block C can be expressed as

$$S = D + P \tag{5.4}$$

$$C = D' + P \tag{5.5}$$

where the P is the predicted block, D is the residual block and D' is the reconstructed residual block. Then, the SSD can be expressed as

$$SSD = ||S - C||_F^2 = ||D + P - D' - P||_F^2 = ||D - D'||_F^2 \tag{5.6}$$

where $||\,||_F$ is Frobenius norm.

In the procedures above, we know that T_{DCT} denotes the DCT matrix and it is a unitary matrix, which has the following property [20]:

$$||X||_F = ||T_{DCT} X T'_{DCT}||_F \tag{5.7}$$

where, X is a square matrix.

According to Equation (7) we can also express the SSD in transform domain as:

$$||F-F'||_F^2 = ||T_{DCT}DT_{DCT}^T - T_{DCT}D'T_{DCT}^T||_F^2 = ||T_{DCT}(D-D')T_{DCT}^T||_F^2 = ||D-D'||_F^2 = SSD \tag{5.8}$$

Where, F and F' are the transformed residual block and inverse quantized-transformed residual block. The quantization is applied in the transformed coefficients of F. Thus, the cause of the SSD is due to the quantization errors in the integer transformed residual block F'. For example, if quantization step = 18 and all the transform coefficients are multiple of 18, then quantized coefficients could be same as the unquantized coefficients as

$$F = \begin{bmatrix} 180 & 72 & 54 & 18 \\ 72 & 36 & 18 & 18 \\ 36 & 18 & 18 & 18 \\ 18 & 18 & 18 & 0 \end{bmatrix}$$

and

$$F' = \begin{bmatrix} 180 & 72 & 54 & 18 \\ 72 & 36 & 18 & 18 \\ 36 & 18 & 18 & 18 \\ 18 & 18 & 18 & 0 \end{bmatrix}$$

This demonstrated that even most of the coefficients of F are large, SSD can be zero since$||F-F'||_F^2 = 0$. That means a large difference of original block and predicted block may not lead to a large SSD. Similarly, a large SSD does not mean that the predicted block is quite different from original one. Besides the high computational requirement of SSD, SSD cannot also provide very decisive information for mode decision. Therefore, SSD is not a good choice for block mode classification. Similarly, R_{motion} is the number of bits for encoding the motion vector, which is also not highly related to final mode decision.

The reason why H.264 brings in variable block-size modes is that in some cases current MB cannot be well predicted by large modes, which means the difference between current MB and predicted MB is quite large. Thus, we have to resort to small modes for better estimation. The $R_{residue}$ is the number of bits after entropy coding of residue blocks. It is directly related to the difference between original MB and predicted MB. Thus, $R_{residue}$ for large block size such Inter 16×16 mode could be a good measure for classifying motion content of the MB for fast mode selection. For example, if $R_{residue}$ of Inter 16×16 mode is small, which means that the large block-size mode can well predict the current MB, then it is likely that the current MB should be a SMB and the best mode is among large block-size modes; otherwise, we should focus on small block-size modes to determine the best mode. Based on this idea, we choose the bit-rate for encoding the Inter 16×16 residue block $R_{Inter16\times16}$ as the measure for SMB and CMB classification and then using different mode searching orders to save some of unnecessary modes computation. The classification can be based on a pre-defined threshold as follows:

1. If $R_{Inter16\times16} < Threshold$, then the current MB is SMB;
2. If $R_{Inter16\times16} \geq Threshold$, then the current MB is CMB;

In order to demonstrate that $R_{Inter16\times16}$ could provide very good classification accuracy, the classification accuracy of six (06) QCIF sequences with 50 frames is shown in Table 5.1 with quantization parameter (QP) = 30 and threshold = 140. The Correct Ratio is the ratio of which the MB classification accords with the result obtained from the exhaustive mode selection, SMB Error Ratio is the ratio of which SMBs are mistakenly categorized into the CMBs, and the opposite ratio is called CMB Error Ratio. From the Table 5.1, we can find that when the thresholds are properly chosen, the classification correct ratio is very high, which can be up to 98%.

5.2.2 Fast Inter 16×16 Rate-based Mode Selection Algorithm

Based on the $R_{Inter16\times16}$ for motion content classification, we can use different mode searching orders to avoid some unnecessary modes calculation for achieving fast mode selection. The proposed fast Inter 16×16 rate-based mode decision algorithm first calculates the $R_{Inter1616}$ of current MB, and then compares it with a pre-defined threshold. If the $R_{Inter16\times16}$ is less than the threshold, the current block is determined as a SMB which covers large modes; otherwise, it is a CMB which covers small modes. The fast algorithm can be summarized as:

- Step 1: Compute the bit-rate for the Inter16x16 mode residue block $R_{Inter16\times16}$
- Step 2: If $R_{Inter16\times16} <$ Threshold select the best mode from DIRECT/SKIP, Inter $R_{Inter16\times16}$, Inter 8×16 and Inter 16×8 modes based their rate-distortion costs.
- Step 3: If $R_{Inter16\times16} \geq$Threshold old then select the best mode from Inter P 8×8 (Inter 8×4, Inter 4×8, Inter 4×4), Intra 16×16 and Intra 4×4 modes based their rate-distortion costs.

Table 5.1 Macroblock Motion Content Classification Accuracy based on $R_{Inter16\times16}$

Sequences	CMB Error Ratio	SMB Error Ratio	Correct Ratio
Akiyo	1.02%	0.40%	98.58%
Foreman	5.71%	0.60%	93.69%
Carphone	7.32%	0.57%	92.11%
Stefan	11.63%	6.17%	82.20%
Trevor	12.09%	1.56%	86.35%
Salesman	5.43%	0.30%	94.27%

The flow chart of the proposed fast mode decision algorithm is shown in Fig.5.1. With use of this search orders, a lot of computation can be saved due to unnecessary and computational intensive modes especially the intra modes calculation can be avoided. In addition, the proposed algorithm is very easy to implement in both hardware and software and it does not require extra module because the calculation of $R_{Inter16\times16}$ is a necessary step in the process of rate-distortion cost computation. In the proposed algorithm as shown in Fig.5.1, the reason why we select the Intra 16×16 mode in the CMB path is that when the prediction accuracy of Inter 16×16 mode is good enough, we do not have to resort to Intra 16×16mode, which is only a compensation of large block-size modes.

To efficiently apply the proposed algorithm, we have to resolve how to select good threshold values for different *QP*s, which may have great influence on the computation time and rate-distortion performance. The classification accuracy and rate-distortion coding performance of the proposed fast algorithm using various *QP*s and threshold values are shown in Table 5.2. From the table, we can find that the larger the *QP* values, the larger the correct ratio, which results in a very similar coding performance between original algorithm and the proposed fast algorithm. In addition, the RDO performance is not very sensitive to the selection of thresholds. Take $QP = 34$ as an example; as long as we choose a threshold value in the zone of [90,150], the RDO performance is quite stable. This property is significant since we do not have to exert much effort on how to precisely determine the best thresholds. With extensive simulations on different kinds of video sequences and *QP*s, we formulated the relationship between the threshold and *QP*s as following equation:

$$Threshold(QP) = 0.36QP^2 - 34P + 838.6 \tag{5.9}$$

From the above expression, it is easy to determine that when *QP* increases, the threshold reduces. The rule is reasonable since when *QP* increases, the quantization step increases which leads to smaller $R_{Inter16\times16}$. Thus, we have to decrease the threshold to adjust to the smaller $R_{Inter16\times16}$.

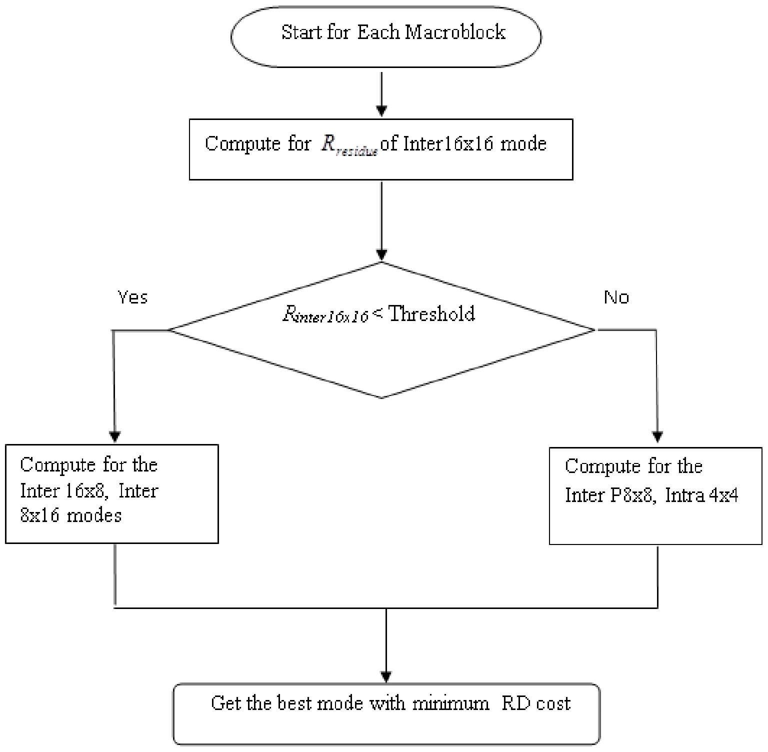

Start for Each Macroblock
Compute for $R_{residue}$ of Inter16x16 mode
$R_{Inter16x16}$ < Threshold
Yes
No
Compute for the Inter 16x8, Inter 8x16 modes
Compute for the Inter P8x8, Intra 4x4
Get the best mode with minimum RD cost

Fig. 5.1 Flow chart of proposed algorithm.

5.2.3 A Fast Intra 4×4 Mode Selection Algorithm

Intra prediction means that the prediction of current macroblock comes from the spatial information of already encoded macroblock in the same image [74]. It mainly contributes to the I-frame encoding and is also an important prediction way in P-frame and B-frame when inter prediction cannot work well. In Intra 16×16 mode, there are four prediction modes: Vertical, Horizontal, DC and Plane prediction [75]. The intra 16×16 prediction works well in a gently changing area. In Intra 4×4 mode, there are totally nine prediction modes supported as shown in Fig. 5.2. Eight prediction modes are for a specific prediction direction and one mode is DC prediction mode. In H.264 standard, a full search (FS) is used to examine all nine modes for the Intra 4×4 prediction to find the one with the smallest RD cost as the best mode.

Therefore, for luminance components in a macroblock, we have to examine $16 \times 9 = 144$ different RD cost calculations; while in FS, inter prediction requires seven RD cost

Table 5.2 Classification Correct Ratio and Coding Performance using different threshold

QP	Threshold	Correct Ratio of classification	PSNR(dB)	Bit rate (bpp)	Original PSNR(dB)	Original Bit rate (bpp)
22	170	73.08%	40.28	562.51	40.31	558.36
	200	73.13%	40.27	562.30		
	230	73.00%	40.27	562.92		
26	120	84.03%	37.39	329.64	37.43	328.29
	150	84.14%	37.38	330.17		
	180	83.84%	37.38	330.66		
30	100	93.60%	34.78	204.99	34.83	205.31
	130	93.75%	34.78	205.17		
	160	93.72%	34.78	205.46		
34	90	98.14%	32.34	133.54	32.35	133.31
	120	98.31%	32.34	133.44		
	150	98.26%	32.33	133.52		

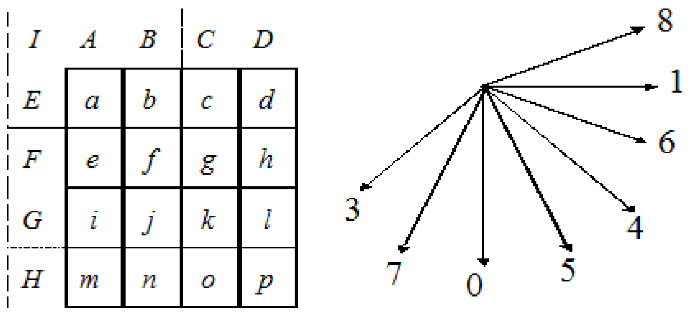


Fig. 5.2 Nine intra prediction modes.

calculations. From the comparison, we can find that the computation load on Intra 4×4 mode is quite large, since there are too many modes to cover. Therefore, if we can exclude some unnecessary prediction modes in advance, much computation time can be saved. To accelerate the coding process, the JVT reference software version JM 6.1d provides a fast SAD-based cost function:

$$J_{SAD} = SAD + \lambda_1.4K \tag{5.10}$$

where *SAD* is sum of absolute difference between the original block S and the predicted block P, and the K equal to 0 for the probable mode and 1 for the other modes. The *SAD* is expressed by

$$SAD = \sum_{i=1}^{4} \sum_{j=1}^{4} |S_{ij} - P_{ij}| \tag{5.11}$$

Where S_{ij} and P_{ij} are the (*i,j*)th elements of the current original block S and the predicted block P, respectively. This *SAD*-based cost function could save a lot of computations as the distortion part is based on the differences between the original block and the predicted block instead of the reconstructed block. Thus, the processes of image block transformation, quantization, inverse quantization, inverse transformation and reconstruction of image block can all be saved. In addition, the number of bits is estimated by constants either equal 4 or 0. Thus, the variable length coding using CAVLC or CABAC can also be saved. However, the expense of the computation reduction usually comes with quite significant degradation of coding efficiency. To achieve better rate-distortion performance, JM6.1d also provided an alternative *SATD*-based cost function:

$$J_{SATD} = SATD + \lambda_1.4K \tag{5.12}$$

where, *SATD* is sum of absolute Hadamard-transformed [76] [77] difference between the original block *S* and the predicted block *P*, which is given by

$$SATD = \sum_{i=1}^{4} \sum_{j=1}^{4} |h_{ij}| \tag{5.13}$$

where, h_{ij} are the (*i,j*)th element of the Hadamard transformed image block H which is the difference between the original block *S* and the predicted block *P*. The Hadamard transformed block H is defined as

$$H = T_H(S-P)T_H^T \tag{5.14}$$

With

$$T_H = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}$$

Experimental results show that the J_{SATD} could achieve better rate-distortion performance than the J_{SAD}, but it requires more computation due to the Hadamard transformation. How-

Table 5.3 Probability of real intra mode among K modes in sense of SATD criterion

Sequence	QP	K=2	K=3	K=4
Akiyo	28	0.7569	0.8402	0.8965
	32	0.7501	0.8311	0.8761
	36	0.7327	0.8188	0.8596
	40	0.7691	0.8634	0.9125
Foreman	28	0.7456	0.8276	0.8761
	32	0.7587	0.8462	0.8846
	36	0.7634	0.8501	0.8969
	40	0.7402	0.8336	0.8791
Stefan	28	0.7621	0.8491	0.8935
	32	0.7763	0.8652	0.9063
	36	0.7532	0.8340	0.8834
	40	0.7669	0.8558	0.8967

ever, the performance degradation of *SATD* is still large; therefore, a new Intra 4×4 mode fast selection algorithm is proposed.

In many cases, the best intra prediction mode found by full search is not the same as that from *SATD* criterion; nevertheless, it is likely that the real best mode is among the smallest K modes in the sense of *SATD* criterion and we define such probability as P, as shown in Table 5.3. From the table, we can find that when K = 3, the probability P is up to 0.85. Based on this result, we propose a fast Intra 4×4 mode selection algorithm as follows:

- Step1: Calculate the SATD cost of nine Intra 4×4 prediction modes;
- Step2: Single out K modes with smallest SATD cost which are collected in the set ϕ ;
- Step3: Calculate the RD cost among those modes in the set ϕ and determine the best prediction mode.

In the proposed algorithm, the computation complexity of SATD is much less than that of RD cost and we merely need to examine K modes instead of full search. Therefore, the computation time can be reduced quite a lot.

5.3 Simulation Results

The proposed mode selection algorithm have been tested using the first 50 frames from three video sequences (Akiyo, Foreman and Stefan) all in QCIF format 176×144. They correspondingly present different kinds of video: slow motion, medium motion and fast motion. The experiment was carried out in the JVT JM 12.0 encoder.

Table 5.4 Computation time saving, PSNR reduction and Bit rate for Akiyo

(a) Computation time saving in Akiyo(%)

QP	Threshold	Fast Inter	Fast Inter + Fast Intra(K=2)	Fast Inter + Fast Intra(K=3)	Fast Inter + Fast Intra(K=4)
28	169	-64.23%	-71.44%	-70.72%	-69.24%
32	120	-63.41%	-70.61%	-69.42%	-68.45%
36	81	-63.20%	-70.02%	-68.95%	-67.68%
40	55	-62.55%	-69.66%	-68.33%	-67.37%

(b) PSNR reduction in Akiyo

QP	Threshold	Fast Inter	Fast Inter + Fast Intra(K=2)	Fast Inter + Fast Intra(K=3)	Fast Inter + Fast Intra(K=4)
28	169	-64.23%	-71.44%	-70.72%	-69.24%
32	120	-63.41%	-70.61%	-69.42%	-68.45%
36	81	-63.20%	-70.02%	-68.95%	-67.68%
40	55	-62.55%	-69.66%	-68.33%	-67.37%

(c) Bit rate increase saving in Akiyo

QP	Threshold	Fast Inter	Fast Inter+Fast Intra(K=2)	Fast Inter+Fast Intra(K=3)	Fast Inter+Fast Intra(K=4)
28	169	-0.15%	+0.69%	+0.43%	+0.34%
32	120	-0.10%	+0.31%	+0.65%	+0.33%
36	81	+0.03%	+1.04%	+0.58%	+0.28%
40	55	+0.15%	+1.06%	+0.88%	+0.77%

The test parameters are listed as below:

- Number of previous frames used for inter motion search is 1;
- CABAC is enabled;
- GOP structure is IPPPIPPP . . . ;
- Max search range is 32;
- QP values are 28, 32, 36 and 40, and the corresponding thresholds are calculated with experience formula: 55, 81, 120,169.

Table 5.5 Computation time saving, PSNR reduction and Bit rate for Foreman

(a) Computation time saving in Foreman(%)

QP	Threshold	Fast Inter	Fast Inter + Fast Intra(K=2)	Fast Inter + Fast Intra(K=3)	Fast Inter + Fast Intra(K=4)
28	169	-63.58%	-71.67%	-70.34%	-69.19%
32	120	-63.77%	-71.21%	-69.95%	-68.86%
36	81	-63.49%	-70.59%	-69.81%	-68.33%
40	55	-63.18%	-70.00%	-69.07%	-67.89%

(b) PSNR reduction in Foreman

QP	Threshold	Fast Inter	Fast Inter + Fast Intra(K=2)	Fast Inter + Fast Intra(K=3)	Fast Inter + Fast Intra(K=4)
28	169	-0.06	-0.06	-0.06	-0.05
32	120	-0.05	-0.04	-0.03	-0.03
36	81	-0.01	-0.01	0.00	-0.01
40	55	0.00	0.00	0.00	0.00

(c) Bit rate increase saving in Foreman(%)

QP	Threshold	Fast Inter	Fast Inter + Fast Intra(K=2)	Fast Inter + Fast Intra(K=3)	Fast Inter + Fast Intra(K=4)
28	169	+0.02%	+1.50%	+0.87%	+0.57%
32	120	-0.14%	+1.76%	+0.75%	+0.58%
36	81	+0.06%	+2.37%	+1.58%	+1.33%
40	55	+0.01%	+1.20%	+0.73%	+0.24%

- In the fast intra mode selection algorithm, we make experiments in the case of K = 2, 3, 4.

Compared with the original H.264/AVC encoder with the rate-distortion optimization and computation time, the proposed algorithm achieves large computation time reduction on average as well as quite similar R-D performance, as listed in Table 5.4 (for the sequence of Akiyo), Table 5.5 (for the sequence of Foreman) and Table 5.6 (for the sequence of Stefan). When only fast inter mode selection algorithm is employed, in Akiyo and Foreman, the average computation reduction can reach 62% or more, with average 0.04 PSNR reductions and average 0.2% bitrate increase and in Stefan, the average computation reduction reaches 47%, with average 0.02 PSNR reductions and average 0.12% bitrate increase. When we combine fast inter mode and intra mode selection algorithms together in the case of K=2, 3 and 4, the computation time can be further reduced, with a little degradation of R-D

Table 5.6 Computation time saving, PSNR reduction and Bit rate for Stefan

(a) Computation time saving in Stefan(%)

QP	Threshold	Fast Inter	Fast Inter + Fast Intra(K=2)	Fast Inter + Fast Intra(K=3)	Fast Inter + Fast Intra(K=4)
28	169	-46.34%	-57.19%	-55.05%	-51.75%
32	120	-51.66%	-62.48%	-61.27%	-59.19%
36	81	-54.10%	-64.46%	-62.69%	-60.53%
40	55	-56.51%	-65.23%	-63.79%	-62.18%

(b) PSNR reduction in Stefan

QP	Threshold	Fast Inter	Fast Inter + Fast Intra(K=2)	Fast Inter + Fast Intra(K=3)	Fast Inter + Fast Intra(K=4)
28	169	-0.05	-0.13	-0.10	-0.07
32	120	-0.05	-0.14	-0.11	-0.10
36	81	-0.06	-0.09	-0.05	-0.04
40	55	-0.03	-0.03	-0.03	-0.04

(c) Bit rate increase saving in Stefan(%)

QP	Threshold	Fast Inter	Fast Inter + Fast Intra(K=2)	Fast Inter + Fast Intra(K=3)	Fast Inter + Fast Intra(K=4)
28	169	+0.01%	+0.31%	+0.15%	+0.11%
32	120	+0.14%	+0.37%	+0.02%	-0.20%
36	81	-0.39%	+0.15%	-0.08%	-0.21%
40	55	+0.01%	+0.61%	+0.24%	+0.28%

Table 5.7 R-D performance of skip mode early termination algorithm for Akiyo, Foreman and Stefan

QP	PSNR reduction			Bit-rate increase			Saved Time		
	Akiyo	Foreman	Stefan	Akiyo	Foreman	Stefan	Akiyo	Foreman	Stefan
28	0.00	+0.01	0.00	+1.29%	+1.48%	+0.35%	-52%	-11%	-6%
32	-0.01	0.00	0.00	+1.34%	+1.33%	+0.57%	-60%	-23%	-12%
36	-0.03	-0.04	-0.01	+0.35%	+1.11%	+0.44%	-70%	-36%	-21%
40	-0.01	-0.11	-0.03	+1.85%	+1.63%	+0.74%	-72%	-47%	-32%

performance. The results shown in the tables indicate that K=3 can lead to a better tradeoff between computation load and R-D performance than K = 2 or 4.

From the simulation results, we can find that:

Table 5.8 R-D performance of fast intra mode decision algorithm (JVT-G013) for Akiyo, Foreman and Stefan

QP	PSNR reduction			Bit-rate increase			Saved Time		
	Akiyo	Foreman	Stefan	Akiyo	Foreman	Stefan	Akiyo	Foreman	Stefan
28	-0.11	-0.03	-0.05	+1.13%	+0.83%	+1.05%	-25%	-25%	-28%
32	+0.01	-0.04	-0.07	+3.31%	+1.88%	+0.92%	-24%	-24%	-26%
36	-0.05	-0.01	-0.07	+2.56%	+1.40%	+1.39%	-23%	-23%	-25%
40	+0.01	+0.01	-0.09	+3.31%	+2.33%	+1.38%	-23%	-21%	-23%

Table 5.9 R-D performance of fast inter mode selection algorithm (JVT-I020) for Akiyo, Foreman and Stefan

QP	PSNR reduction			Bit-rate increase			Saved Time		
	Akiyo	Foreman	Stefan	Akiyo	Foreman	Stefan	Akiyo	Foreman	Stefan
28	-0.05	-0.06	-0.02	+0.45%	+0.93%	+0.26%	-47%	-27%	-19%
32	-0.03	-0.05	-0.02	+0.64%	+0.42%	+0.10%	-44%	-26%	-17%
36	-0.04	0	-0.01	+0.28%	+0.59%	+0.12%	-41%	-25%	-16%
40	-0.04	-0.04	+0.01	-0.82%	+0.21%	+0.22%	-38%	-22%	-15%

1. The threshold values determined by equation (5.2) are widely adapted to different kinds of video sequences: slow motion, medium motion and high motion. This also indicates that the thresholds are not sensitive to QP values.

2. The proposed algorithm can also reduce more computation time in slow motion sequences than in higher motion sequences. That is because there are more large mode macroblocks in slow motion sequences, so that much computation on small modes can be skipped, especially intra modes, which are quite computationally intensive.

3. When QP value decreases, the degradation of R-D performance increases. From Table 5.2 we can find that in large QP values, the correct ratio of classification is higher, which determines the degradation of R-D performance. Therefore, a smaller QP leads to a smaller correct ratio of classification and worse R-D performance.

Table 5.7, 5.8 and 5.9 show the R-D performance and computation time of three conventional fast mode selection algorithms: skip mode early termination algorithm, fast intra mode decision algorithm and fast inter mode decision algorithm. The results indicate that the proposed algorithm has a better tradeoff between R-D performance and computation load.

5.4 Summary

In this chapter, rate based efficient fast inter and intra mode selection algorithms for H.264/AVC standard are proposed. In the fast inter mode algorithm, the motivation is to skip some unlikely modes by dividing the macroblock into CMB and SMB based on the bit-rate for encoding the Inter 16×16 mode residue block $(R_{Inter16\times16})$. The threshold value used for the classification is also formulated by an equation in terms of QPs, which can maintain a good tradeoff between computation complex and rate-distortion performance. The contribution of fast intra mode algorithm is to examine three most probable intra prediction modes rather than full search. Experimental results demonstrate that the proposed algorithms achieve large computation savings while maintaining the very similar rate-distortion performance.

Chapter 6

RELATIVE SAD-BASED INTER MODE DECISION ALGORITHM FOR H.264/AVC

In this chapter, a new relative sum of absolute difference (RSAD) based inter mode decision algorithm is proposed for H.264/AVC video standard. The main idea is to determine the best inter mode based on RSAD cost instead of the rate-distortion cost. This approach can avoid most of the computationally intensive processes in the H.264/AVC mode decision, such as integer transform, quantization, variable length coding and pixel-reconstruction. Besides, it can solve the problem of SAD-based cost function for H.264/AVC mode decision in biasing to the smaller block size modes, as they are normally achieving higher prediction accuracy. It is demonstrated by experimental results that the proposed RSAD-based mode decision algorithm can reduce 60% to 68% of the H.264/AVC total encoding time with negligible degradation in the rate-distortion performance. The proposed algorithm is also combined with SKIP mode early termination algorithm for further time saving and computation reduction. The enhanced algorithm with SKIP mode early termination technique could save up to 10% more of the encoding time with only a little rate-distortion degradation.

6.1 Challenges and Literature Survey

The newest video coding standard is known as H.264/AVC [18], [78], which greatly outperforms the previous MPEG-1/2/4 and H.261/263 [79]-[81] standards in terms of both picture quality and compression efficiency. To achieve this superior coding performance, H.264/AVC adopts many advanced techniques, such as directional spatial prediction for intra frame coding, variable and hierarchical block transform, arithmetic entropy coding, multiple reference frame motion compensation, etc. It also uses seven different block sizes $(16 \times 16, 16 \times 8, 8 \times 16, 8 \times 8, 8 \times 4, 4 \times 8 \textit{ and } 4 \times 4)$ for motion compensation in inter mode coding, and two different block sizes with various spatial directional prediction modes in intra modes. The purpose to employ variable block size coding is that large block modes (such as $16 \times 16, 16 \times 8 \textit{ and } 8 \times 16$) can be used for stationary image block prediction with high coding efficiency while small block modes can be used for high or complex image blocks with better prediction accuracy.

Although the variable block size technique can significantly improve the prediction accuracy, it also introduces heavy computation loading. In order to select the best macroblock (MB) coding mode, H.264/AVC encoder needs to compute the rate-distortion costs of all possible modes [82][83], which involves computation of integer transform, quantization, variable length coding and pixel-reconstructions processes. All of these processes explain the high computational complexity of rate-distortion cost calculation. The intensive computation greatly limits the application of H.264/AVC in real-time and portable device; therefore, many fast mode decision algorithms, which can reduce the computational complexity without compromising coding efficiency, have been proposed for real-time implementation of H.264/AVC encoders.

In the literature, a number of fast mode decision algorithms [84]-[93] have been proposed to reduce the computational complexity of H.264/AVC. In [84], Jing et al. uses the mean absolute frame difference of the current frame and mean absolute difference of the current MB to determine whether the current MB belongs to homogeneous regions or not and thus only check $16 \times 16, 16 \times 8 \textit{ and } 8 \times 16$ modes. Similarly, in [85], Wu et al. uses the spatial homogeneity and the temporal stationarity characteristics of video objects to select only a small number of inter modes in the rate-distortion optimization process. The MB's edge intensity is applied to detect the spatial homogeneity and collocated MB difference is used to decide the stationarity. By doing this some unnecessary modes are not checked and computation loading is reduced. Kim et al. proposes an adaptive mode decision algorithm [86] by using the property of an all-zero coefficients block to effectively skip unnecessary modes. In [87]-[89], Kuo et al, Yang et al and Fan et al proposes a mode decision algorithm that determines the best mode according to the motion field distribution and correlation within a MB. In [90], Gao and Lu propose a mode selection algorithm to skip some unlikely modes using the rate-distortion cost gaps between different modes. In [91] a scalable fast mode selection algorithm is proposed. In [92], Wei et al. proposes a mode prediction algorithm that can predict the possible modes of current MB using the modes of adjacent MBs. Then a mode checking technique is used to early terminate the mode decision process with an adaptive threshold. Low complexity based transform and quantization approach is given in [93].

Some of the aforementioned fast mode decision algorithms try to classify the MB into large partition or small partition and skip checking some unnecessary modes [94]. However, they still need to compute the rate-distortion costs of some possible modes for the ultimate mode decision, which involve computationally intensive processes of image transformation, quantization, entropy coding, and pixel reconstruction. In H.264/AVC video coding standard, SAD-based and SATD-based cost functions are developed as fast mode decision techniques for avoiding these computationally intensive processes. However, the major drawback is that the rate-distortion performance of the encoded video is quite degraded, which affects their practical implementation. In this chapter, a relative sum of absolute value (RSAD) based fast mode decision algorithm is proposed, which can avoid rate-distortion cost computation while maintaining high rate-distortion performance of the H.264/AVC codec.

6.2 Rate Distortion Cost Function for H.264/AVC

In H.264/AVC encoding process, the best MB coding mode is selected by computing the rate-distortion cost of all possible modes. The best mode is the one with minimum rate-distortion cost and this cost is defined as [78][18].

$$J_{RD}(S,C,Mode|QP) = SSD(S,C,Mode|QP) + \lambda.R(S,C,Mode|QP) \tag{6.1}$$

where, QP is the quantization parameter, λ is the Lagrangian multiplier. A strong connection between the local Lagrangian multiplier and the QP was found experimentally:

$$\lambda = 0.852^{(QP-12)/3} \tag{6.2}$$

The R in (1) can be written as

$$R = R_{header} + R_{motion} + R_{esidual} \tag{6.3}$$

where, R_{header}, R_{motion} and $R_{residual}$ present the number of bits need to encode the header information, motion vectors and quantized residual block, respectively.

In equation (6.1), MODE is chosen from the set of potential prediction modes as below:

For Intra frame:

MODE $\in$ {Intra 4×4, Intra 16×16} and for Inter frame:

MODE $\in$ {SKIP, Inter16×16, Inter 16×8, Inter 8×16, Inter $P8 \times 8$, Intra 4×4, Intra 16×16 }

SKIP mode means that no motion or residual bits will be encoded (only the mode indicator is transmitted). Inter P8×8 mode actually contains four sub-modes. A similar rate-distortion cost function is employed to determine the optimal sub-mode that achieves minimal rate-distortion cost. Once the best sub-mode is found, the minimal cost for the MB is evaluated by searching all the possible modes.

Inter $P8 \times 8 \in$ { Inter 8×8, Inter 8×4, Inter 4×8, Inter 4×4}

SSD(S, C) is the sum of the squared difference (SSD) between the original blocks S and the reconstructed block C, and it can be expressed as:

$$SSD(S,C) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1}(S_{ij} - C_{ij})^2 = \|S - C\|_F^2 \tag{6.4}$$

where S_{ij} and $C_{ij}$are the (i,j)th elements of the current original block S and the reconstructed block C, respectively. Moreover, N is the image block size (N = 4 in H.264 standard) and $\| \ \|_F$ is Frobenius norm. It is also found that the computation of spatial-domain SSD is very time-consuming, since we have to obtain the reconstructed block after Transformation $\rightarrow$ Quantization $\rightarrow$ Inverse Quantization $\rightarrow$ Inverse Transformation $\rightarrow$ Pixel Reconstruction for each possible prediction mode [94]. Besides, encoding bits R is also difficult to calculate since it needs entropy coding, such as CAVLC or CABAC to obtain, which is also quite a time-consuming process. To accelerate the coding process, the JVT reference software provides a fast SAD-based cost function:

$$J_{SAD} = \begin{cases} SAD(S,P) + \lambda_1.4K & \text{for intra } 4 \times 4 \text{ mode;} \\ SAD(S,P) & \text{otherwise.} \end{cases} \tag{6.5}$$

where, SAD(S,P) is sum of absolute difference between the original block S and the predicted block P. The λ_1 is also approximate exponential function of the quantization parameter (QP) which is almost the square of λ, and the K equal to 0 for the probable mode and 1 for the other modes. This SAD-based cost function could save a lot of computations as the distortion part is based on the differences between the original block and the predicted block instead of the reconstructed block. Thus, the processes of image block transformation, quantization, inverse quantization, inverse transformation and reconstruction of image block can all be saved.

In addition, the number of bits is estimated by constants either equal 4 or 0. Thus, the variable length coding using CAVLC or CABAC can also be skipped. However, the expense of the computation reduction usually comes with quite significant degradation of coding efficiency. The rate-distortion performance comparison of H.264/AVC using RDO-based and SAD-based cost functions for different QPs is shown in above in Table 6.1. Compared with RDO-based encoder, the performance degradation of SAD-based cost functions are not-ignorable. Thus, SAD is not an appropriate criterion to determine the best mode. Actually, the SAD values of different inter modes usually have the following relations:

$SAD_{inter16\times16} \geq SAD_{inter16\times8} \geq SAD_{inter8\times16} \geq SAD_{inter8\times8}$

which is shown in Table 6.2. The reason behind is that the small partition motion estimation can always provide better prediction accuracy than the large mode motion estimation. According to this relationship, the J_{SAD} cost function is inclined to choose smaller block mode as the best mode since their SAD values are smaller. Thus, it can explain why J_{SAD} cost function is not appropriate for the best mode decision.

Table 6.1 R-D Performance Comparison Using RDO-based and SAD-based Cost Function in terms of PSNR (dB) and Rate (kbps)

Sequence	QP	28		32		36		40	
		RDO	SAD	RDO	SAD	RDO	SAD	RDO	SAD
Akiyo	PSNR	38.76	38.56	35.85	35.63	33.14	32.97	30.59	30.48
	Rate	153.17	157.54	104.98	109.20	71.27	73.76	49.15	51.73
Foreman	PSNR	36.13	35.92	33.50	33.22	31.00	30.66	28.52	28.18
	Rate	248.51	257.17	155.01	162.55	99.95	105.21	66.36	71.29
Stefan	PSNR	34.96	34.67	31.55	31.20	28.52	28.11	25.66	25.33
	Rate	593.41	602.52	370.54	376.56	228.97	232.76	141.70	145.20

Table 6.2 The probability of SAD relationship among different inter modes

Sequences	$P(SAD_{16\times16} \geq SAD_{16\times8})$	$P(SAD_{16\times16} \geq SAD_{8\times16})$	$P(SAD_{16\times16} \geq SAD_{8\times8})$	$P(SAD_{8\times16} \geq SAD_{8\times8})$
Akiyo	88.70%	83.65%	79.44%	86.24%
Foreman	79.70%	82.95%	86.91%	83.63%
Stefan	85.83%	84.55%	86.43%	89.11%

6.3 Review of the SKIP Mode Early Termination Algorithm

H.264/AVC, SKIP mode is employed to encode some static objects and background that almost remain the same in the adjacent frames [94]. Many real-world video sequences contain substantial amounts of SKIP modes. Table 6.3 shows the probability of the SKIP mode occurring in P frames. The results imply that the SKIP mode usually dominates among other modes. The motion vector in SKIP mode is predicted by using the motion vectors of the neighboring MBs. Besides, no residual block is encoded; therefore, the transform, quantization and entropy coding can be avoided. When the SKIP mode is selected, only a skip indicator is transmitted. Thus, if we can decide in advance whether the best mode is SKIP mode or not, the computationally intensive mode decision process can be omitted. Previous work has contributed a lot to the SKIP mode prediction. According to the work, SKIP mode is the best mode as long as the following four conditions are satisfied:

1. The best motion compensation block size for this MB is 16×16.
2. The best reference frame is the previous frame.
3. The best motion vector is the predicted motion vector (PMV).
4. The transform coefficients of the 16×16 block size are all quantized to zero.

Table 6.3 The probability of SKIP mode in different sequences

QP	28	32	36	40
Akiyo	86.01%	88.29%	92.19%	95.82%
News	76.91%	80.99%	84.66%	89.99%
Foreman	33.79%	45.09%	54.87%	65.24%
Container	82.37%	87.97%	91.09%	96.32%
Stefan	21.44%	29.06%	34.94%	47.02%
Mobile	4.64%	13.31%	26.03%	46.56%

When a MB satisfies these four conditions, it is safe to say that the best mode is SKIP mode. However, in practical use, we can only check conditions (2)-(4) and it may mispredict MBs as skipped. Table 6.4 shows the rate-distortion degradation due to the SKIP mode early termination algorithm without detecting the condition (1). The results indicate that the prediction accuracy is good enough without the assumption of condition (1). In the following section, the RSAD based mode decision algorithm is proposed and it will be combined with SKIP mode early termination algorithm to further improve the coding efficiency.

Table 6.4 Rate-Distortion degradation using SKIP mode early termination algorithm

Sequence	28		32		36		40	
	$\triangle PSNR - dB$	$\triangle Bit\ Rate\%$	$\triangle PSNR - dB$	$\triangle Bit\ Rate\%$	$\triangle PSNR - dB$	$\triangle Bit\ Rate\%$	$\triangle PSNR - dB$	$\triangle Bit\ Rate\%$
Akiyo	-0.02	-0.33	-0.02	-0.29	-0.01	-0.08	0	+0.07
News	-0.01	-0.13	-0.02	-0.27	-0.01	-0.15	0	+0.05
Foreman	-0.03	-0.48	-0.02	-0.37	-0.01	-0.16	0	+0.12
Container	-0.01	-0.12	-0.02	-0.52	-0.02	-0.39	0	+0.08
Stefan	-0.01	-0.10	-0.02	-0.24	-0.03	-0.36	-0.01	-0.09
Mobile	-0.01	-0.15	-0.03	-0.65	-0.02	-0.54	-0.01	0.16

6.4 Proposed RSAD based Mode Decision Algorithm

6.4.1 Definition of RSAD Based Cost Function

The reason why H.264/AVC employs variable block-size modes is that sometimes the current MB cannot be well predicted by large partitions, which means the difference between the current MB and predicted MB is quite large. Thus, the MB should be divided into smaller partitions for better motion compensation. On the other hand, if the current block can be well predicted by the large block mode, it is unnecessary to encode the block with smaller partitions which require more encoding complexity and more bits to represent the motion vectors and side information. Thus, we need to use an appropriate criterion to tell whether the current MB achieves adequate prediction accuracy or not. SAD value presents the difference between the current MB and predicted MB, so it can reflect the prediction accuracy of different modes. As mentioned in previous section, however, J_{SAD} cost function is not appropriate for mode decision since smaller modes usually lead to smaller SAD values so that smaller block mode is inclined to be considered as the best mode. Therefore, we should admit the intrinsic SAD differences between small partitions and large partitions. Besides, we should realize that the SAD values are also significant in mode decision. A small SAD difference among different modes indicates that large partition may well predict the current MB without splitting into small partitions; while a large SAD difference means the prediction accuracy of small modes outperforms that of the large mode, so that the small mode is likely to be the best mode. Since the SAD difference covers a wide range and fluctuates greatly, it is difficult to claim whether the SAD difference is small or large. Therefore, we can define a new parameter, relative SAD (RSAD) cost, to measure the SAD difference between two modes. The RSAD cost between two modes, mode 1 (representing large partition) and mode 2 (representing small partition), is defined as below:

$$RSAD_{1\rightarrow 2} = \frac{(SAD_1 - SAD_2)}{SAD_1} \tag{6.6}$$

where, $RSAD_{1\rightarrow 2}$ represents the relative prediction accuracy between mode 1 and mode 2. According to the definition of RSAD, a small RSAD cost indicates a small SAD difference between two modes, which means that their prediction accuracy is similar. In this case, mode 1 is likely to be a better choice since it needs fewer bits to encode its motion vectors. On the other hand, a large RSAD cost implies that mode 2 is more likely to be the better mode since its prediction accuracy obviously outperforms the large partition mode 1. Therefore RSAD is a new measure to select the better mode.

6.4.2 RSAD Cost Function for Mode Decision

RSAD cost is a new measure to select the better mode instead of the conventional costs such as SAD, SATD and RD cost. It represents the SAD difference which can measure the prediction accuracy difference between two modes. Based on this idea, a threshold T_R, is employed to tell whether the RSAD cost is small or large. If $RSAD_{1\rightarrow 2} < T_R$, select the large partition mode 1 as the better mode; Else select the small partition mode 2 as the better mode.

It is obvious that T_R affects the probability of mode 1 or mode 2 to be the winning mode. When T_R increases, mode 1 is more likely to be the better mode; when T_R reduces, mode 2 is more likely to be the better mode. Therefore, how to appropriately select the threshold T_R is important to the encoding performance. Note that T_R is related to SAD_1 since it indicates the prediction accuracy of the large partition. If SAD_1 is small, the mode 1 (large partition) is more likely to be the better mode; thus, T_R should be larger. If SAD_1 is large, the mode 2 is more probable to be the better mode; thus, T_R should be smaller. To implement this idea, we can use another threshold T_R to distinguish whether SAD_1 is small or large.

Besides, T_S should be adaptive and related to the quantization parameter (QP) of the H.264 codec. The reason behind is that a higher QP leads to a more rough quantization; thus the error between the original block and reconstructed block also increases, which has a negative influence on the motion estimation performance. Then the overall SAD values will rise, so the threshold T_S should also increase.

We, firstly, introduce the RSAD-based mode decision algorithm for two modes (mode 1 is the larger partition mode) that can be further implemented in H.264/AVC to select the best mode among all the inter modes.

- Step I: Initialize QP value, $T_S, T_{RS}, T_{RL}, T_S, T_{RS}, T_{RL}$.

- Step II: Calculate the *SAD*s of mode 1 and mode 2: SAD_1 and SAD_2.
- Step III: If $SAD_1 < T_S$ then $T_R = T_{RS}$ and $T_R = T_{RS}$; else $T_R = T_{RL}$ and $T_R = T_{RL}$
- Step IV: Calculate $RSAD_{1\to2} = (SAD_1 - SAD_2)/SAD_1$
- Step V: If $RSAD_{1\to2} < T_R$, choose mode 1 as the better mode; Else, choose mode 2 as the better mode. In this algorithm, we need to compare $RSAD_{1\to2}$ with the threshold T_R.

If $RSAD_{1\to2} \leq T_R$, $(CSAD_{s8\times4}, CSAD_{s4\times8}, CSAD_{s4\times4}) \leq T_R$, then $SAD_1 \leq SAD_2 + T_R.SAD_1$. Thus, the comparison between $RSAD_{1\to2}$ and T_R is equal to the comparison between SAD_1 and the compensated SAD_2 (SAD_2 plus T_R SAD_1.

6.4.3 RSAD Based Cost Implemented for H.264/AVC Inter Mode Decision

International video coding standard named H.264/AVC supports seven inter modes of different block size and shapes, including mode 16×16, 16×8, 8×16, $P8\times8$, (8×8, 8×4, 4×8 and 4×4) which is shown in Fig.6.1. In this research, $RSAD_{16\times16\to16\times8}$, $RSAD_{16\times16\to8\times16}$, and $RSAD_{16\times16\to P8\times8}$ are employed for mode decision and we need to find the best mode among these four inter modes. In order to present the proposed algorithm clearly, compensated SAD (CASD) values of mode 16×8, 8×16 and $P8\times8$ are employed in the following algorithms.

The RSAD-based mode decision algorithm for H.264/AVC can be implemented as shown in below:

- Step I: Initialize QP value, $T_{S1}, T_{RS1}, T_{RL1}, T_{S2}, T_{RS2}, T_{RL2}$.
- Step II: Calculate the SAD cost of four sub inter modes: $SAD_{s16\times16}, SAD_{s16\times8}, SAD_{s8\times}, SAD_{s8\times8}$.
- Step III: If $SAD_{s16\times16} < T_{S1}$ then $T_{R1} = T_{RS1}$ and $T_{R2} = T_{RS2}$; else $T_{R1} = T_{RL1}$ and $T_{R2} = T_{RL2}$
- Step IV: Calculate compensated $CSAD_{s16\times8}, CSAD_{s8\times16}, CSAD_{s8\times8}$
  with $CSAD_{s16\times8} = SAD_{s16\times8} + T_{R1}.SAD_{s16\times16}$
  $CSAD_{s16\times8} = SAD_{s8\times16} + T_{R1}.SAD_{s16\times16}$
  $CSAD_{s8\times16} = SAD_{s8\times16} + T_{R1}.SAD_{s16\times16}$
  $CSAD_{s8\times8} = SAD_{s8\times8} + T_{R1}.SAD_{s16\times16}$

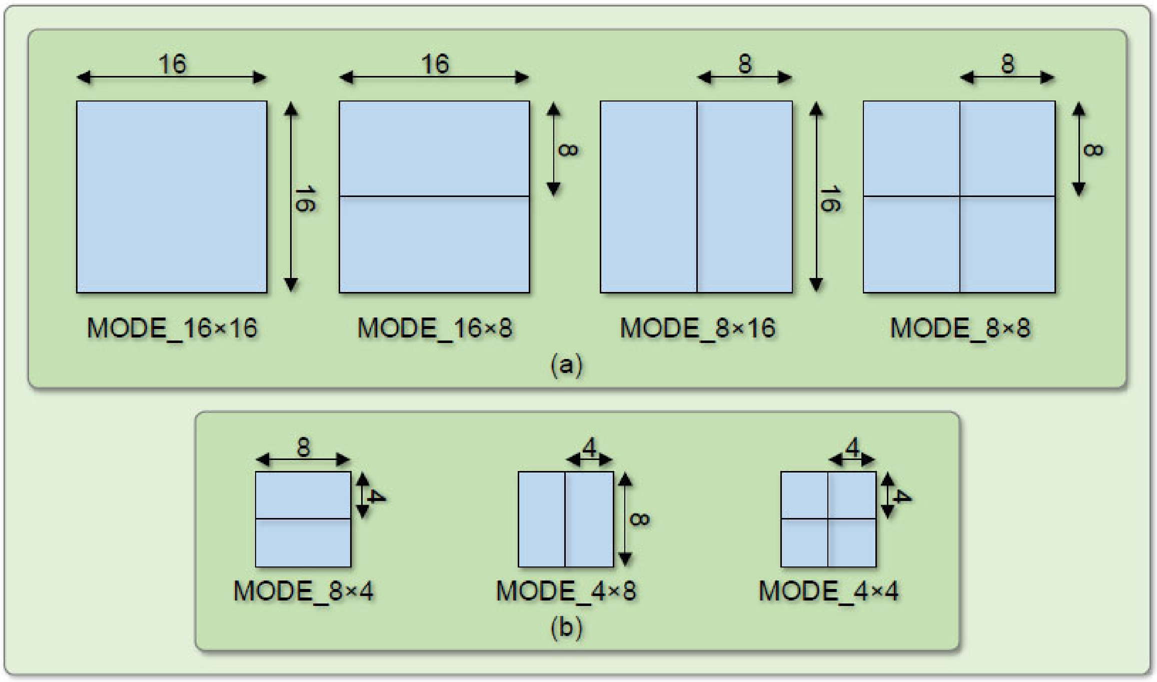


Fig. 6.1 Seven inter modes of different block size and shapes in H.264/AVC.

- Step V: Best mode = argmin $CSAD_{s16\times16}$, $CSAD_{s16\times8}$, $CSAD_{s8\times16}$, $CSAD_{s8\times8}$

Based on this algorithm, we can determine the best inter mode by comparing the compensated SAD cost instead of calculating the rate-distortion cost for each inter mode. In many cases, it is unnecessary to search intra modes for inter frame block coding because experimental results given in Table 6.5 show that the RD performance improvement induced by intra modes is trivial and the neglect of intra modes can save a lot of time and reduce computation complexity drastically.

In the algorithm above, the RSAD cost has been implemented to select the best mode from four Inter modes. Actually, in regard to the Inter $P8\times8$ mode, it contains four sub-modes: Inter $S8\times8$, Inter $S8\times4$, Inter $S4\times8$ and Inter $S4\times4$ modes.

In H.264/AVC standard, for each 8×8 block in a MB, rate-distortion cost function is employed to determine the best sub-mode. Similar to the case of large partition, we can implement the proposed RSAD-based mode decision algorithm to choose the best sub-mode among all possible sub-modes (small partition).

- Step I: Initialize QP value, $T_{S3}, T_{RS3}, T_{RL3}, T_{S4}, T_{RS4}, T_{RL4}$.

- Step II: Calculate the SAD cost of four sub inter modes: $SAD_{s8\times8}, SAD_{s8\times4}, SAD_{s4\times8}, SAD_{s4\times4}$.

Table 6.5 The probability of intra mode in inter frames

QP	28	32	36	40
Akiyo	0	0	0	0
News	0	0	0	0
Foreman	0	0	0.05%	0.05%
Container	0	0	0	0
Stefan	0.55%	0.14%	0.32%	0.46%
Mobile	0.20%	0.11%	0	0

- Step III: If $SAD_{s8\times8} < T_{S3}$ then $T_{R3} = T_{RS3}$ and $T_{R4} = T_{RS4}$; else $T_{R3} = T_{RL3}$ and $T_{R4} = T_{RL4}$
- Step IV: Calculate compensated $CSAD_{s8\times4}, CSAD_{s4\times8}, CSAD_{s4\times4}$
  with $CSAD_{s8\times4} = SAD_{s8\times4} + T_{R3}.SAD_{s8\times8}$
  $CSAD_{s8\times4} = SAD_{s4\times8} + T_{R3}.SAD_{s8\times8}$
  $CSAD_{s4\times8} = SAD_{s4\times8} + T_{R3}.SAD_{s8\times8}$
  $CSAD_{s4\times4} = SAD_{s4\times4} + T_{R3}.SAD_{s8\times8}$
- Step V: Best mode = argmin $CSAD_{s8\times8}$, $CSAD_{s8\times4}$, $CSAD_{s4\times8}$, $CSAD_{s4\times4}$

where T_{S3}, T_{RS3}, and T_{RL3} are SAD threshold and RSAD thresholds for selecting Inter sub modes of $s8\times4$ and $s4\times8$ partitions and T_{S4}, T_{RS4}, and T_{RL4} are SAD threshold and RSAD thresholds for selecting Inter sub mode of $s4\times4$. The flow chart of the proposed RSAD-based mode decision algorithm is shown in Fig.6.2.

6.5 Parameters Selection

As mentioned earlier that the selection of the RSAD thresholds T_R and thresholds for classifying small and large SAD of the large partition is vital for the proposed RSAD-based mode decision algorithm. So, how to select those thresholds such that they are not sensitive to different video content is a vital task. So, how to select those thresholds such that they are not sensitive to different video content is a vital task. In order to evaluates the best thresholds for the proposed algorithm, at first we do not consider the sub-mode related parameters and only consider T_{RS1}, T_{RS2}, T_{RL1}, T_{RL2} and T_S. Here, we use the iterative searching method to obtain the optimal thresholds for the proposed algorithm as following:

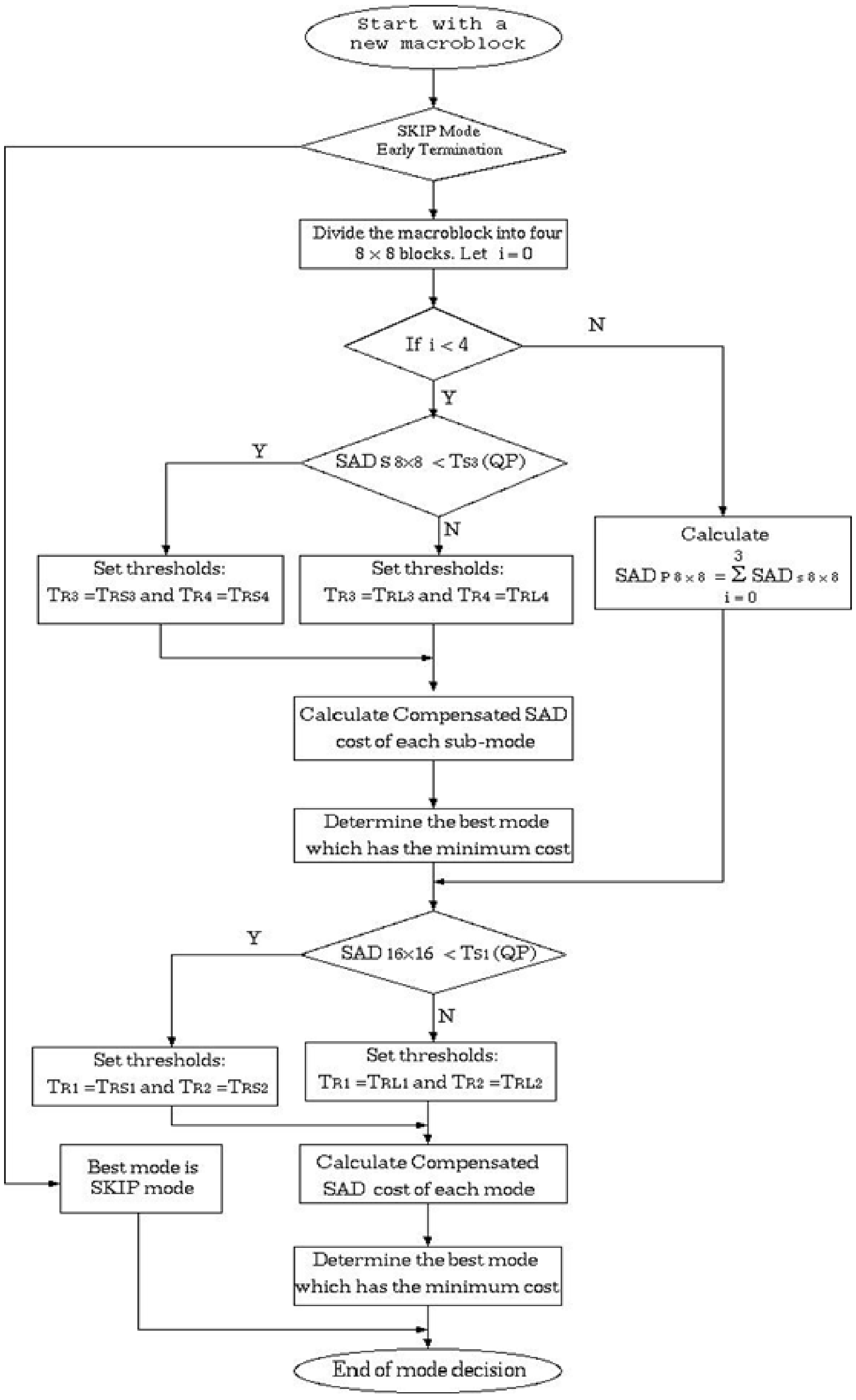

Start with a new macroblock
SKIP Mode Early Termination
Divide the macroblock into four 8 × 8 blocks. Let i = 0
If i < 4
N
Y
Y
SAD S 8×8 < Ts3 (QP)
N
Set thresholds: TR3 =TRS3 and TR4 =TRS4
Set thresholds: TR3 =TRL3 and TR4 =TRL4
Calculate SAD P 8×8 = Σ SAD s 8×8 i = 0 3
Calculate Compensated SAD cost of each sub-mode
Determine the best mode which has the minimum cost
Y
SAD 16×16 < Ts1 (QP)
N
Set thresholds: TR1 =TRS1 and TR2 =TRS2
Set thresholds: TR1 =TRL1 and TR2 =TRL2
Best mode is SKIP mode
Calculate Compensated SAD cost of each mode
Determine the best mode which has the minimum cost
End of mode decision

Fig. 6.2 Flow diagram of the proposed RSAD based mode decision algorithm.

- Step 1: Initialize the threshold $T_{RS1}(0)$, $T_{RS2}(0)$, $T_{RS3}(0)$,$T_{RS4}(0)$ and$T_{RS5}(0)$ for a specific QP value;
- Step 2: Adjust the thresholds T_{RS1}, T_{RS2}, T_{RS3}, and T_{RS4} in order to achieve local minimum RD cost. Denote the updated threshold as $T_{RS1}(i)$, $T_{RS2}(i)$, $T_{RS3}(i)$, $T_{RS4}(i)$;
- Step 3: Adjust the threshold T_{S1} in order to achieve smaller RD cost. Denote the updated threshold as $T_{S1}(i)$;
- Step 4: Repeat Step 2 and Step 3 until the RD cost cannot be further reduced. Record the optimal values.

Based on the extensive simulation upon various types of video contents (low motion, medium motion and high motion), we find that the selection of the thresholds T_{RS1}, T_{RS2}, T_{RL1}, and T_{RL2} is nearly not varied with the video contents as shown in Table 6.6 and the relationship between T_{S1} and QP is also very robust, which is shown in Table 6.7. In addition, the optimal thresholds T_{RS3}, T_{RS4}, T_{RL3}, and T_{RL4} for sub-mode selection can also be obtained via the iterative searching method. These results are shown in Tables 6.8 and Table 6.9.

Table 6.6 The Optimal thresholds T_{RS1}, T_{RS2}, T_{RL1} and T_{RL2} found based on simulation

Qp	16				24			
	T_{RS1}	T_{RS2}	T_{RL1}	T_{RL2}	T_{RS1}	T_{RS2}	T_{RL1}	T_{RL2}
Akiyo	0.07	0.15	0.30	0.03	0.15	0.30	0.03	0.07
Foreman	0.06	0.15	0.30	0.03	0.15	0.30	0.03	0.07
Stefan	0.07	0.16	0.30	0.03	0.15	0.30	0.03	0.07
Qp	28				32			
	T_{RS1}	T_{RS2}	T_{RL1}	T_{RL2}	T_{RS1}	T_{RS2}	T_{RL1}	T_{RL2}
Akiyo	0.15	0.30	0.03	0.06	0.15	0.30	0.03	0.07
Foreman	0.15	0.30	0.03	0.06	0.15	0.31	0.03	0.07
Stefan	0.14	0.28	0.03	0.05	0.16	0.31	0.03	0.06
Qp	36				40			
	T_{RS1}	T_{RS2}	T_{RL1}	T_{RL2}	T_{RS1}	T_{RS2}	T_{RL1}	T_{RL2}
Akiyo	0.16	0.30	0.03	0.06	0.15	0.30	0.03	0.07
Foreman	0.15	0.30	0.03	0.07	0.15	0.30	0.03	0.07
Stefan	0.15	0.30	0.03	0.07	0.15	0.30	0.03	0.08

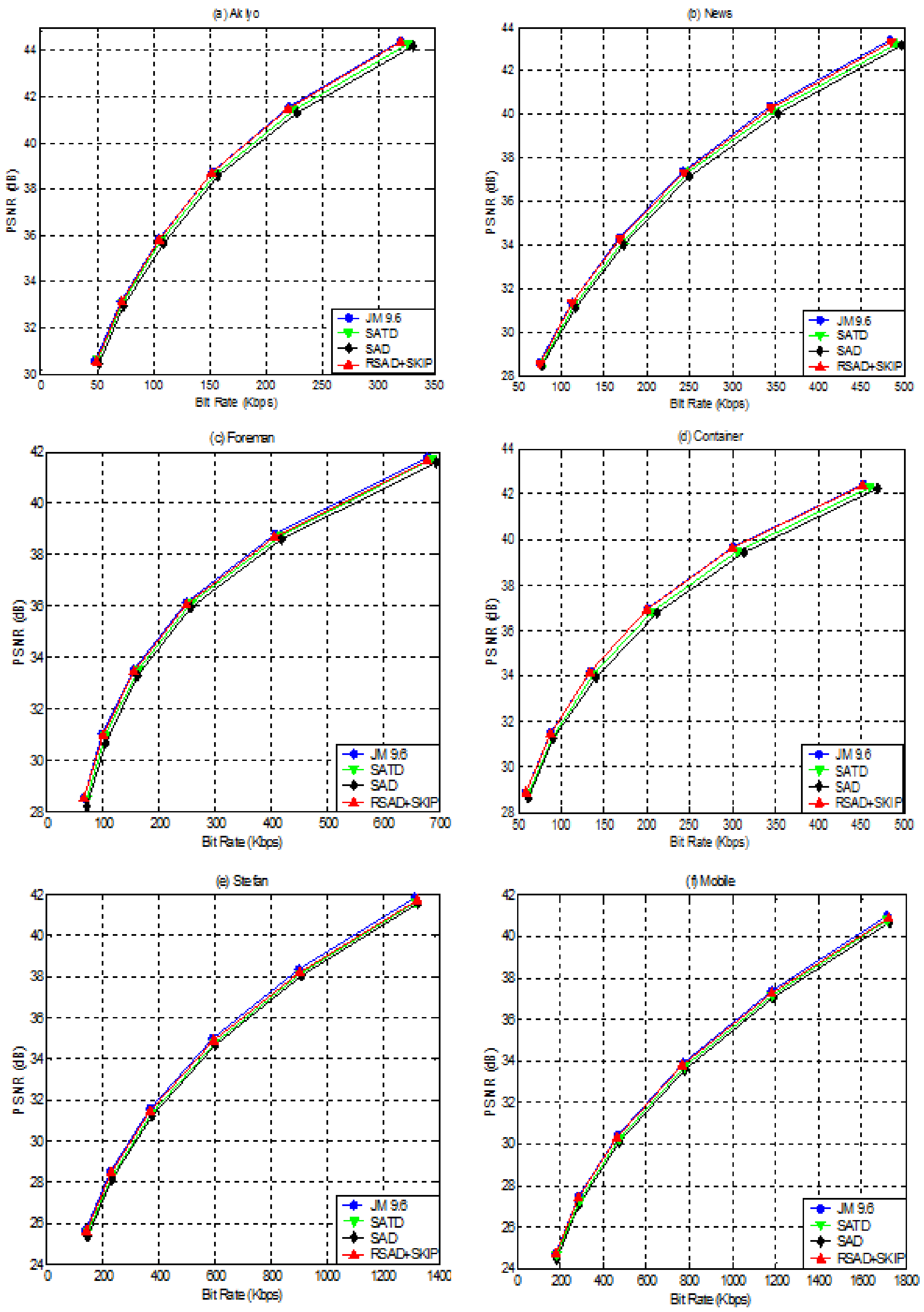


Fig. 6.3 Comparison of Rate-Distortion performance among JM 9.6 standard SAD- based, SATD-based and proposed RSAD based cost function combined with SKIP mode early termination algorithm.

Table 6.7 The optimal threshold T_{S1} found based on simulation

Sequence	QP = 16	QP = 24	QP = 32	QP = 40
Akiyo	200	400	900	1550
Foreman	200	400	850	1500
Stefan	200	400	900	1500

Table 6.8 The optimal thresholds T_{RS3}, T_{RS4}, T_{RL3} and T_{RL4} found based on simulation

Qp	20				24			
	T_{RS3}	T_{RS4}	T_{RL3}	T_{RL4}	T_{RS3}	T_{RS4}	T_{RL3}	T_{RL4}
Akiyo	0.20	0.40	0.10	0.20	0.20	0.40	0.10	0.20
Foreman	0.20	0.40	0.10	0.20	0.20	0.40	0.10	0.20
Stefan	0.20	0.40	0.10	0.20	0.20	0.40	0.10	0.20
Qp	32				40			
	T_{RS3}	T_{RS4}	T_{RL3}	T_{RL4}	T_{RS3}	T_{RS4}	T_{RL3}	T_{RL4}
Akiyo	0.20	0.40	0.10	0.20	0.20	0.40	0.10	0.20
Foreman	0.20	0.40	0.10	0.20	0.20	0.40	0.10	0.20
Stefan	0.19	0.40	0.10	0.20	0.20	0.40	0.10	0.20
Qp	32				40			
	T_{RS3}	T_{RS4}	T_{RL3}	T_{RL4}	T_{RS3}	T_{RS4}	T_{RL3}	T_{RL4}
Akiyo	0.20	0.40	0.10	0.21	0.20	0.40	0.10	0.20
Foreman	0.20	0.40	0.10	0.21	0.20	0.40	0.10	0.20
Stefan	0.20	0.40	0.10	0.20	0.20	0.40	0.10	0.20

Table 6.9 The optimal threshold T_{s2} for $P8 \times 8$ sub-modes found based on simulation

Sequence	QP = 16	QP = 24	QP = 32	QP = 40
Akiyo	150	200	450	800
Foreman	150	250	450	800
Stefan	150	200	450	800

6.6 Results and Discussion

The proposed RSAD based mode decision algorithm was tested using the first 50 frames from six video sequences all in QCIF format. “Akiyo” and “News” are sequences of low spatial detail and changes in motion. “Container” has dominant horizontal motion and “Foreman” has medium motion changes with dominant luminance changes. “Stefan” contains panning

Table 6.10 The relationship between Ts and QP

QP	16	24	32	40
T_{S1}	200	400	900	1500
T_{S2}	150	200	450	800

Table 6.11 The selected thresholds for RSAD based mode decision algorithm

T_{RS1}	T_{RS2}	T_{RL1}	T_{RL2}	T_{RS3}	T_{RS4}	T_{RL3}	T_{RL4}
0.15	0.30	0.03	0.07	0.20	0.40	0.10	0.20

Table 6.12 Encoding time reduction of the proposed RSAD algorithm

QP	20	24	28
Akiyo	-67.75%	-67.23%	-65.87%
News	-68.35%	-66.93%	-65.97%
Foreman	-64.66%	-64.38%	-62.95%
Container	-68.36%	-67.99%	-67.08%
Stefan	-65.65%	-64.22%	-63.92%
Mobile	-67.80%	-66.35%	-65.82%
QP	32	36	40
Akiyo	-64.30%	-63.87%	-62.32%
News	-64.43%	-64.06%	-63.22%
Foreman	-61.60%	-61.39%	-60.48%
Container	-66.08%	-63.98%	-62.29%
Stefan	-62.66%	-60.59%	-59.81%
Mobile	-65.02%	-61.25%	-60.68%

motion and has distinct fast motion changes. “Mobile” has slow panning, zooming and a complex horizontal and vertical motion.

The experiment was carried out in the JVT JM 9.6 encoder and the test parameters are listed as below:

- CABAC is enabled;
- GOP structure is IPPP;
- Max search range for motion estimation is 32;

Table 6.13 R-D performance of RSAD-based mode decision algorithm

Sequence	QP	Δ Bit Rate %	Δ PSNR-dB
Akiyo	20	-0.05	-0.05
	24	-0.31	-0.05
	28	-0.44	-0.04
	32	-0.22	-0.03
	36	+0.09	-0.01
	40	-0.08	0
News	20	+0.29	-0.05
	24	+0.33	-0.05
	28	+0.15	-0.04
	32	+0.28	-0.03
	36	+0.56	-0.02
	40	+0.46	-0.01
Foreman	20	+0.17	-0.09
	24	-0.07	-0.08
	28	+0.02	-0.07
	32	-0.04	-0.06
	36	+0.10	-0.04
	40	+0.72	-0.02
Container	20	-0.20	-0.04
	24	-0.38	-0.04
	28	-0.55	-0.02
	32	-0.52	-0.03
	36	-0.18	-0.01
	40	+0.05	-0.01
Stefan	20	+0.82	-0.09
	24	+0.50	-0.11
	28	+0.35	-0.10
	32	-0.02	-0.10
	36	+0.35	-0.06
	40	+0.65	-0.03
Mobile	20	+0.42	-0.11
	24	+0.11	-0.09
	28	-0.45	-0.12
	32	-1.21	-0.13
	36	-0.68	-0.07
	40	+0.14	-0.03

Table 6.14 Encoding time reduction of the proposed RSAD algorithm combined with SKIP mode early termination algorithm

QP	20	24	28	32	36	40
Akiyo	-72.43%	-72.01%	-71.73%	-71.57%	-71.45%	-69.93%
News	-72.30%	-72.17%	-71.95%	-71.38%	-71.27%	-71.15%
Foreman	-66.48%	-65.46%	-65.43%	-65.49%	-64.30%	-64.96%
Container	-71.10%	-71.77%	-72.24%	-72.87%	-72.75%	-72.43%
Stefan	-66.11%	-65.26%	-65.13%	-64.92%	-64.92%	-63.80%
Mobile	-68.20%	-67.02%	-66.67%	-66.63%	-66.40%	-66.51%

- QP values are 20, 24, 28, 32, 36 and 40;

These optimal thresholds obtained based on experimental results are also nearly not varying with the different kinds of video contents different QP values. Fortunately, we can select a group of optimal thresholds for $T_{RS1}, T_{RS2}, T_{RL1}, T_{RL2}$,. and $T_{RS3}, T_{RS4}, T_{RL3}, T_{RL4}$. For practical implementation of the proposed RSAD-based mode decision algorithm, which are shown in Table 6.10. Moreover, a set of T_{S1} and T_{S2} thresholds are also selected based on the results of Tables 6.7 and Table 6.9 for difference QPs, which are shown in Table 6.11. In practical implementation, we can use a table to store all the optimal T_{S1} and QP and use table-lookup method to decide those thresholds for a specified QP value.

Compared with the original H.264/AVC encoder in terms of the rate-distortion optimization and computation time, the proposed algorithm achieves a considerable computation time reduction shown in Table 6.12 with a little R-D performance degradation, as listed in Table 6.13. The graphical representation of the encoding time reduction with the proposed RSAD based scheme and combined with SKIP mode early termination scheme are given in Fig. 6.4 and Fig.6.5. From the simulation results, we can conclude that the proposed RSAD algorithm can reduce the encoding time by around 60% to 68% in different QP values. When we combine the proposed algorithm with SKIP mode early termination scheme, the computation time can be further reduced up to 10%, accompanied with a little higher rate-distortion degradation, as shown in Table 6.14. The comparison of rate-distortion performances among JM 9.6 standard, SAD, SATD, and RSAD based mode decision algorithms combined with SKIP mode early termination is shown in Fig.6.3. The result indicates that our proposed algorithm indeed outperforms some other fast algorithm in terms of video quality and compression ratio. The simulation results imply that the combination of SKIP mode early termination algorithm leads to higher computational reduction in slow motion sequences than in high motion sequences. The reason behind is that slow motion sequences contain more static

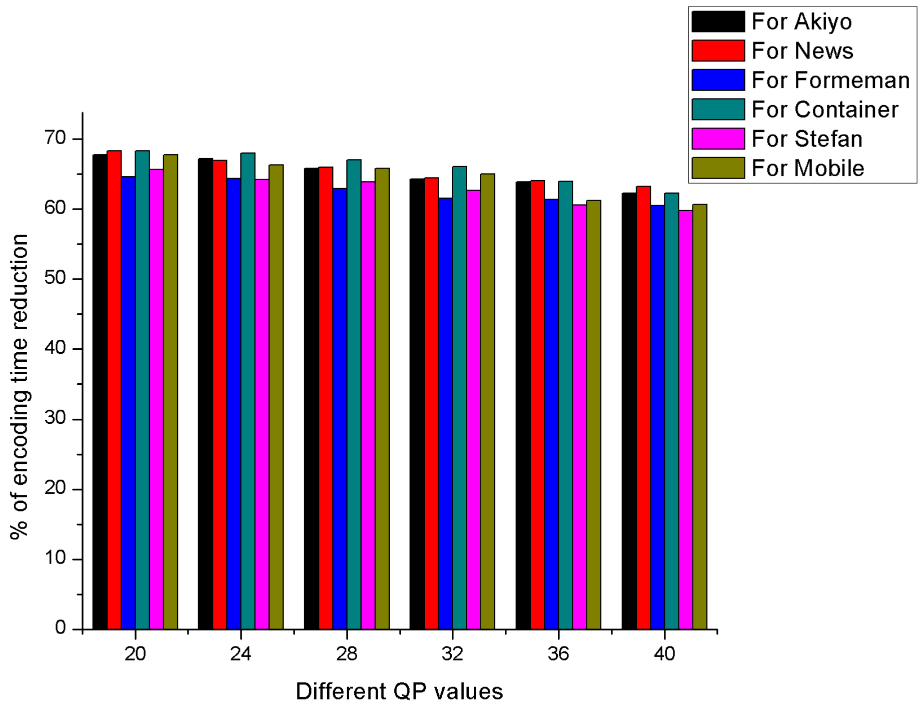


Fig. 6.4 Encoding time reduction of the proposed RSAD algorithm.

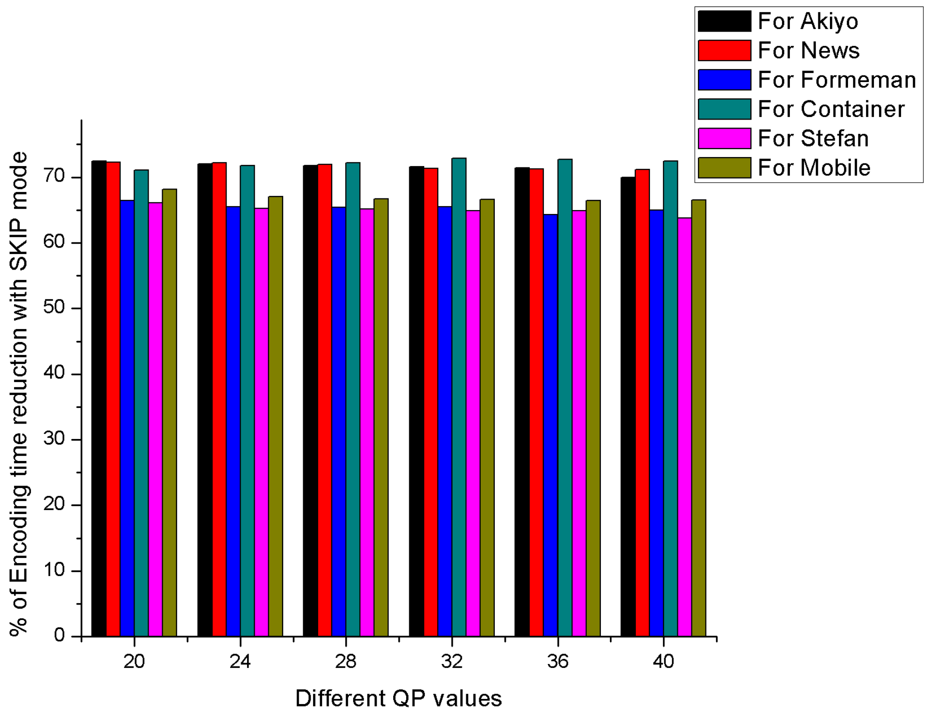


Fig. 6.5 Encoding time reduction of the proposed RSAD algorithm combined with SKIP mode early termination algorithm.

MBs that are appropriate to be encoded by SKIP mode. Similarly, higher coding efficiency can be achieved in the large QP. The encoding time is not the only criterion to evaluate an algorithm's efficiency. In the hardware implementation we mainly focus on the number of rate-distortion operations on different modes because the computation of rate-distortion cost is quite heavy in the coding process. From this aspect, our proposed algorithm is quite efficient since it determines the best mode only based on the proposed RSAD cost function.

Chapter 7

PROBABILITY BASED FAST INTER MODE DECISION ALGORITHM FOR H.264/AVC

In this chapter, a new probability based fast inter mode decision algorithm is proposed for H.264/ AVC video standard. The main idea is to classify the inter modes into large and small modes category and determine which one is suitable for a macroblock (MB). Depending on this, the probability of each large mode corresponding to the initial Hadamard $SAD_{S16\times16}$ cost or the probability of each small mode corresponding to the initial Hadamard $SAD_{S8\times8}$ cost is determined. Then the best mode can be obtained from their probability characteristics. Thus, it is possible to determine the best inter mode only by calculating the Hadamard Transform (H-SAD) cost instead of rate distortion cost. Moreover, it avails the platform to determine the best mode without comparing the cost of each mode to other modes. As a result, this approach can avoid most of the complex computation processes like quantization, variable length coding, pixel reconstruction etc. The experimental results have shown that the proposed algorithm can reduce 60% to 65% of the total encoding time of H.264/ AVC with negligible distortion in the rate distortion performance.

7.1 Literature Review and Challenges

H.264/AVC is the latest video standard which has outperformed previous video compression standard like MPEG-1/2/3 and H.261/263 in terms of video compression efficiency as well as video quality. This is why it has been used for different applications like video streaming, Blue-ray, real time communication etc. It introduces many new techniques which include variable and hierarchical block transform, arithmetic entropy encoding including CABAC and CAVLC, directional spatial prediction for intra frame coding, multiple reference frame motions to outperform the previous standards. Among these features one of the key ones is it uses seven variable blocks sizes for motion estimation and compensation which are 16×16, 16×8, 8×16, 8×8, 8×4, 4×8 and 4×4. Among these, large blocks (16×16, 16×8 and 8×16) are generally used for stationary image prediction with high coding efficiency and small blocks (8×8, 8×4, 4×8 and 4×4) are used for complex images for better prediction accuracy. The large blocks requires less computational time and has less accuracy than the small blocks. On the other hand, small blocks require more computational time and have more accuracy than the large blocks. Therefore, a trade of between the computational time and the prediction accuracy which plays a vital role for video quality is needed to decide which partition block size i.e. mode is the best one for motion estimation of a Macro Block (MB). This vital task is done by computing rate distortion costs of all available blocks and selecting the best partition block size i.e. mode that can provide relatively the best motion compensation with minimum computational time. This rate distortion cost calculation is also a highly complex that includes several processes like computation of integer transform, quantization and arithmetic entropy coding and pixel reconstruction process. This is why this variable block technique involves lots of computational power. As a result, H.264/AVC has limited applications in real time communication.

In [95]-[103] different fast mode algorithms have been proposed applying which it is possible to reduce the computational complexity significantly enabling this video standard for real time application. For examples, in [95], Jing et al proposed mean absolute frame difference of a current frame and mean absolute difference of current MB to determine whether a MB is homogenous or not and decide the best mode among 16×16, 16×8 and 8×16. In [96], Kuo proposes the motion field distribution and correlation method for determining the best modes. In [97], Wei et al, proposes a mode decision algorithm where the mode of a MB is determined using the modes of adjacent MBs. Most of the algorithm proposes methods that skip some unnecessary modes, but need to calculate the rate distortion cost which needs high amount of computational time. Lee et al. in [98] proposed a fast intra mode decision algorithm based on properties of Hadamard transform. In [99], a relative SAD (RSAD) based inter mode decision algorithm is proposed, where it is possible to determine the best mode without calculating rate distortion cost. In [113], Chiang proposes a H.264 encoding method based on statistical learning to reduce the computational cost involved in three main components in H.264 encoder, i.e., intermode decision, multi-reference motion estimation (ME), and intra-mode prediction. Here, the best mode is determined using probability and statistic. In [114] a fast mode selection scheme for H.264/AVC inter prediction based on statistical learning method is proposed to speed up the H.264 encoder with a conventional fast ME algorithm and reduction of the encoding time compared to the original encoder. In [115] Turaga et al. introduce a new measure of block similarity that can improve the inter-intra mode decision significantly. Furthermore, for certain motion-estimation strategies, this measure can also improve the bit rate resulting from a better motion estimation. In [116] an adaptive mode decision algorithm is presented, with rate-distortion optimization that reduces the complexity of the H.264 encoder without loss of image quality and compression ratio. The proposed algorithm uses the property of an all-zero coefficients block that is produced by quantization and coefficient thresholding to effectively skip unnecessary modes. In this research, probability based fast inter mode algorithm is proposed which can decide the best mode without calculating the rate distortion cost maintaining a high rate distortion performance of the H.264/ AVC codec.

7.2 Rate Distortion Cost Function for H.264/AVC Standard

The objective of using variable block size mode in H.264/AVC standard is to ensure precise motion prediction. When a MB cannot be predicted well enough using a large block, it is suitable to divide it into small blocks and thus ensuring better motion compensation. But if it is possible to predict the MB with a large partition, there is no need for using smaller blocks because it introduces higher complexity calculating processing time. The general method to determine suitable partition size for a MB for best motion compensation needs to calculate rate distortion cost using the J_{SAD}, J_{SSD} or similar cost function. For J_{SSD} , sum of square of the difference of current MB and the predicted MB is calculated. In [78], rate distortion cost is define as

$$J_{SSD}(S,C,Mode|QP) = SSD(S,C,Mode|QP) + \lambda .R(S,C,Mode|QP) \tag{7.1}$$

Where, λ is the Lagrangian multiplier and QP is the quantization parameter. The relation between Lagrangian multiplier and QP is found experimentally:

$$\lambda = 0.85 \times 2^{(QP-12)/3} \tag{7.2}$$

The R in (7.1) can be written as $R = R_{header} + R_{motion} + R_{residual}$

Where, R_{header}, R_{motion}, $R_{residual}$ are the number of bits needed to represent the header information, motion vectors and quantized residual block, respectively. SSD (S,C) is the sum of the squared of the difference between the original blocks S and the reconstructed block C and it is defined as:

$$SSD(S,C) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1}(S_{ij} - C_{ij})^2 \tag{7.3}$$

Here, S_{ij} is the *(i, j)* th element of the current block and C_{ij} is the *(i, j)* th element of the reconstructed block. N is the image block size and equal to 4 for H.264. In equation (7.1) best mode is determined from a set of modes mentioned as:

Mode $\in$ { SKIP, Inter 16×16, Inter 16×8, Inter 8×16, Inter 8×8, inter 4×4, inter 16×16 }.

Here, SKIP mode is an especial mode where no residual bit or motion is encoded. For J_{SAD} cost function, SAD value from the difference between the current MB and predicted MB is calculated. Here, J_{SAD} is defined as

$$J_{SAD} = \begin{cases} SAD(S,P) + \lambda_1.4K & ; \\ SAD(S,P) & \text{otherwise.} \end{cases} \tag{7.4}$$

where, *SAD(S,P)* is the sum of absolute difference of the original block S and the predicted block P and defined as

$$SSD(S,C) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1}(S_{ij} - P_{ij}) \tag{7.5}$$

Here, S_{ij} is the *(i,j)*th element of the current block and P_{ij} is the (i,j)th element of the predicted block. K is equal to 0 for probable mode and 1 for other modes and λ_1 is almost the square of λ and almost the exponential function of quantization parameter *QP*.

Between these two, SAD based cost function can save a lot of computation than the *SSD* based cost function because it does not need image reconstruction or variable length coding using CAVLC or CABAC like the *SSD* cost function. But this results a degradation of coding efficiency. To compensate that degradation, sum of absolute transformed differences (SATD) is widely used which uses frequency transform, usually H-SAD of the difference between the original blocks S and the reconstructed block C. It is complex method but takes less computation than SSD based cost function method and also have a better coding efficiency than SAD based cost function. That's why H-SAD cost is considered for the proposed probability based fast inter mode algorithm.

7.3 Review of SKIP Mode Early Termination

In most video sequence, some static and background images almost remain same. No motion or residual bits are encoded for these static portions of a video frame. This is called SKIP mode early termination. In mode of video sequences, substantial amounts of SKIP modes are present. In order to increase the coding efficiency, the H.264 video coding standard introduces the variable block size motion compensation for a better inter frame coding, which adopts seven modes of different block sizes as well as a special mode, the skip mode. Due to the complex nature of the variable block size motion estimation, finding the best inter coding for H.264 would take up a significant complexity. Thus, an efficient skip mode detection out of all inter modes to speed up the encoding process is introduced. In [117], a lot of work has been done to the SKIP mode prediction and states that SKIP mode can be considered as the best mode when the following conditions are satisfied.

1. The best motion compensation block size for this MB is 16×16.
2. The best reference frame is previous frame,
3. The best motion vector is the predicted motion vector.
4. The transform coefficient of the 16×16 block size is all quantized to zero.

The problem is that only the last three conditioned can be checked and this may miss predict the SKIP mode early termination. In the following section proposed probability base fast inter mode decision is explained and it is combined with skip mode to improve the coding efficiency.

7.4 Proposed Probability Based Fast Inter Mode Selection Algorithm

It is seen from observation and rigorous simulations that, when a current MB is homogeneous by nature, the H-SAD cost between the current MB and the predicted MB is very small. And, for a homogeneous MB a large portion block is more suitable than the smaller one for motion prediction. Again, when the MB is non- homogeneous, the H-SAD cost between the current MB and the predicted MB increases depending on the complexity. As the complexity increases, the prediction cost also increases. In these cases, a small partition block is more suitable than a large one for better motion prediction. So, it can be said that when the H-SAD cost of a MB is small, the suitable modes are the large ones and for larger H-SAD cost the suitable modes are the smaller ones.

7.4.1 Determination of Large and Small Modes

As there are seven inter modes of different sizes and shapes including 16×16, 16×8, 8×16, 8×8, 8×4, 4×8 and 4×4 supported by H.264/AVC. There is also a SKIP mode which is employed to encode some static objects and background that remain almost same in adjacent frames. Modes 8×8, 8×4, 4×8 and 4×4 are considered as sub modes and are classified under Inter P 8×8 modes where Inter p $8 \times 8 \in$ {Inter 8×8, Inter 8×4, Inter 4×8, Inter 4×4}. Here we have classified these seven modes into two categories; large modes and Small modes where,

Large modes$\in$ {Inter 16×16, Inter 16×8, Inter 8×16} and
Small modes$\in$ {Inter 8×8, Inter 8×4,Inter 4×8, Inter 4×4}

To find the best mode for a MB, we first calculate the Hadamard $SAD_{16\times16}$ cost of that MB. Then, we divide the MB into four 8×8 sub block and calculate Hadamard $SAD_{s8\times8}$ cost for each block to find Hadamard $SAD_{p8\times8}$ which is defined as

$$SAD_{p8\times8} = \sum_{i=0}^{3} SAD_{s8\times8} \tag{7.6}$$

where, i is the ith sub block of the MB. If the cost Hadamard $SAD_{16\times16}$ is equal or less than $SAD_{p8\times8}$ then we choose the large modes, otherwise we choose the small modes.

7.4.2 Finding the Best Mode from Large and Small Modes

To determine the best mode from large modes we first take the Hadamard $SAD_{16\times16}$ cost of a MB. Then we find out which mode is the best for that cost for general cases. When a set of the Hadamard $SAD_{16\times16}$ costs are taken and their corresponding best mode is plotted in a curve an interesting characteristic of large modes is found. A characteristic curve for "Foreman" QCIF video sequence at different QP values is shown in Fig.7.1. This is a normalized probability curve where it can be seen that a particular mode has a better probability of occurrence at a certain cost. To simplify this thing we divide the data set into small chucks and find out the dominating mode with best probability.

The range at which a certain mode dominates is marked with a threshold value. For three large modes (16×16, 16×8 and 8×16) two thresholds are enough to distinguish among them. To determine the best mode from the small ones Hadamard $SAD_{s8\times8}$ cost is calculated first for every sub-block and the probability of occurrence of each small mode to the value is found. Similar process can be applied to select the best sub mode. As there are four modes, three thresholds values are required to distinguish them. Characteristic curve for small modes of "Foreman" QCIF video sequence is shown in Fig.7.2. Finally, this algorithm is combined with SKIP mode early termination process to make it even faster.

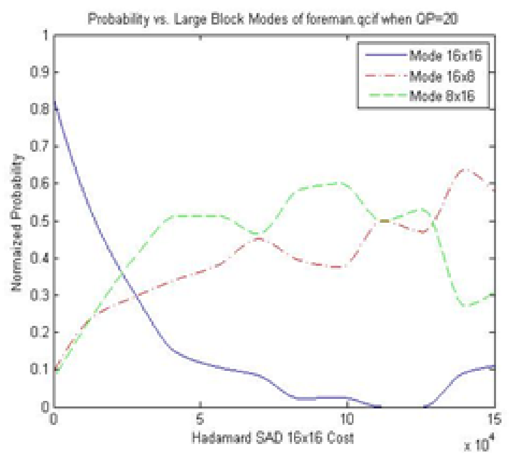
7.4.3 Threshold Determination for Each Mode

There are seven inter modes of different sizes and shapes including 16×16, 16×8, 8×16, 8×8, 8×4, 4×8 and 4×4 supported by H.264/AVC. In this research, these seven block sizes supported by the latest video standard are classified into two categories i.e. Large Blocks (16×16, 16×8 and 8×16) and Small Blocks 8×8, 8×4, 4×8 and 4×4. At first, a MB is taken and its Hadamard $SAD_{16\times16}$ Transform cost calculated. Then the characteristic curve of the probability of large modes vs. Hadamard $SAD_{16\times16}$ Transform cost of that MB is plotted. Finally, the characteristic curve of the probability of small modes vs. In this research, the probability of modes and vs. the initial MB cost characteristic curves of Mobile.qcif video sequence (100 frames) for different quantization parameter (QP = 20, 24, 28, 32, 36, 40) are analyzed. Fig.7.3 and Fig.7.4 shows the characteristic changes between Large Modes and Small modes for different quantization parameters. From large modes probability characteristic curve, it has been observed that the range of domination of 16×16 mode increases as the quantization parameter increases. Similar scenarios are found from the small modes characteristic curves. Therefore, it is necessary to determine some thresholds for each mode to determine their range. But as the range for each mode varies with quantization parameter, the thresholds should be adaptive with quantization parameters.

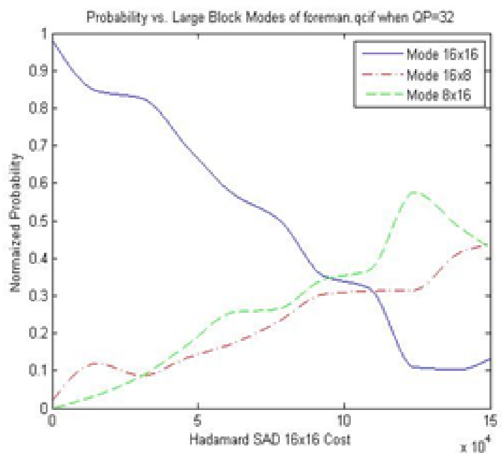
In this research, the probability of modes and vs. the initial MB cost characteristic curves of Mobile.qcif video sequence (100 frames) for different quantization parameter (QP = 20, 24, 28, 32, 36, 40) are analyzed. Fig.7.3 and Fig.7.4 shows the characteristic changes between Large Modes and Small modes for different quantization parameters. From large modes probability characteristic curve, it has been observed that the range of domination of 16×16 mode increases as the quantization parameter increases. Similar scenarios are found from the small modes characteristic curves. Therefore, it is necessary to determine some thresholds for each mode to determine their range. But as the range for each mode varies with quantization parameter, the thresholds should be adaptive with quantization parameters.

7.4.4 Parameters Selection

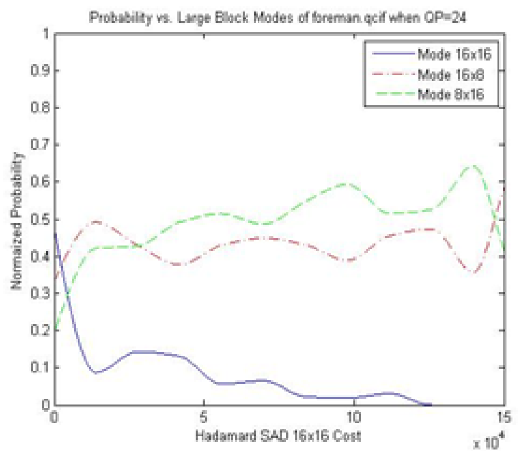
In this research, the proposed algorithm first determines whether the large modes are suitable or the small modes for a MB. Then, the large mode from the large modes category or the small mode from small mode category is selected as the best mode. As there are three large



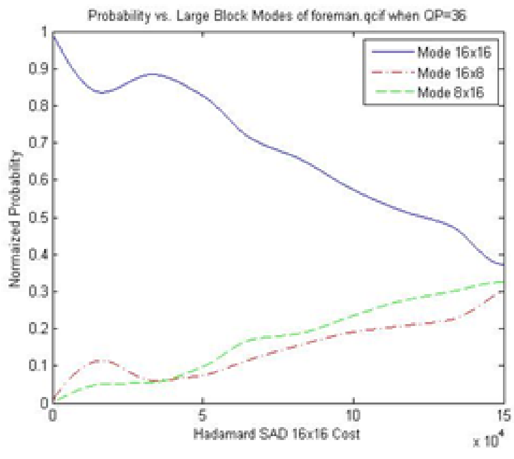
(a)



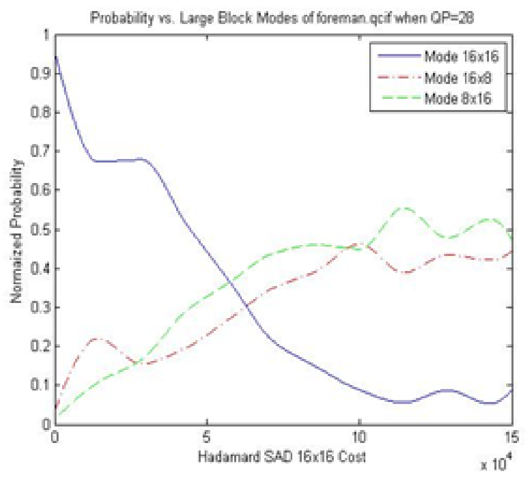
(d)



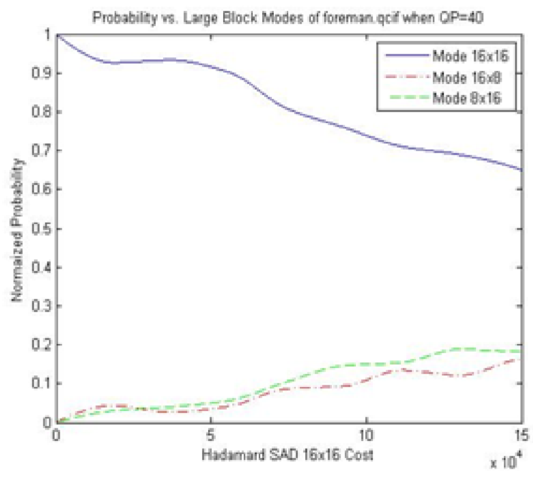
(b)



(e)



(c)



(f)

Fig. 7.1 Probability Characteristic Curve of Large Modes for Foreman QCIF Sequence at QP = 20, 24, 28, 32, 36 & 40.

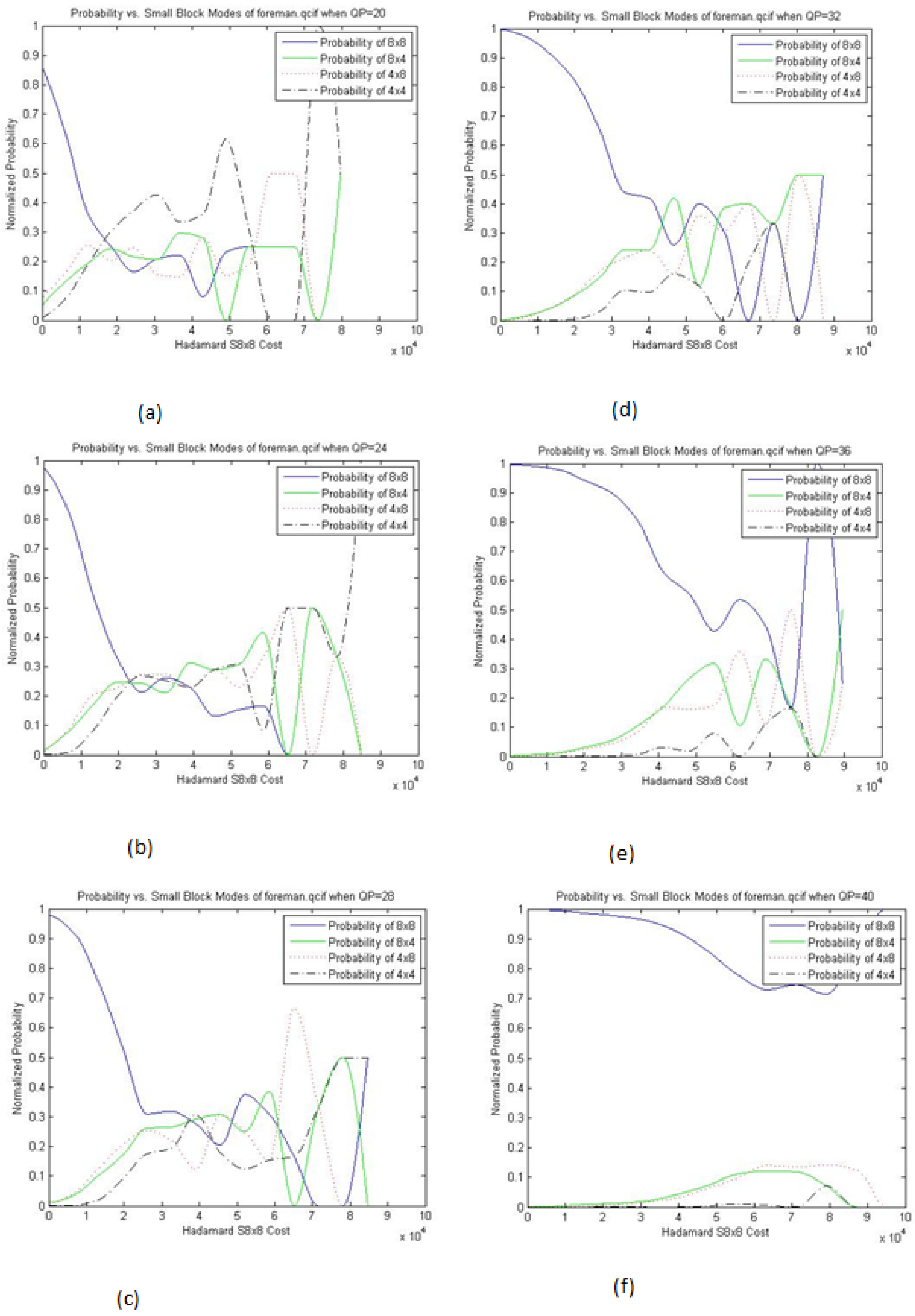
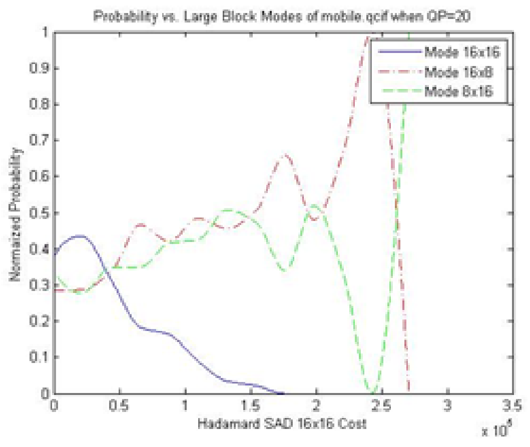
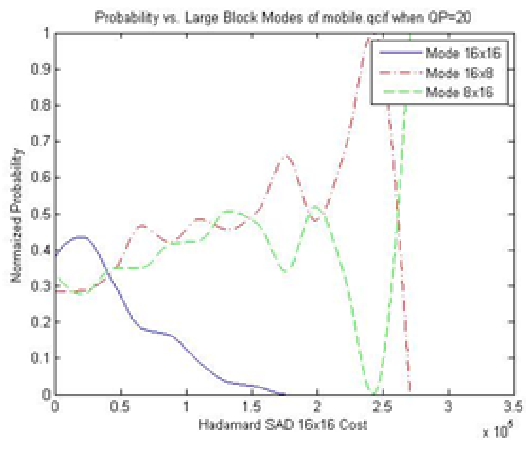


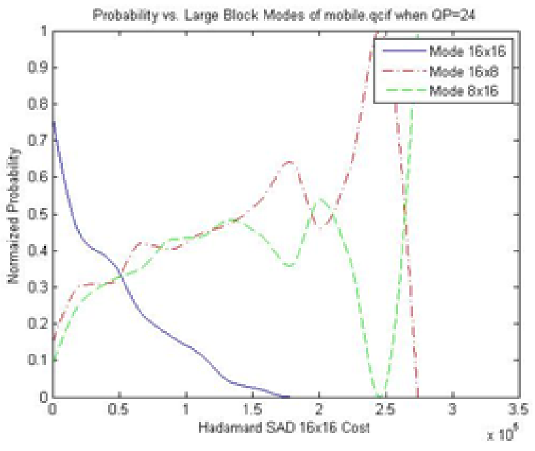
Fig. 7.2 Probability Characteristic Curve of Small Modes for Foreman QCIF Sequence at QP = 20, 24, 28, 32, 36 & 40.



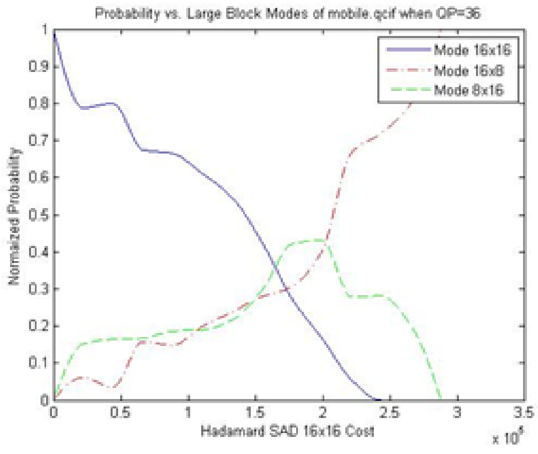
(a)



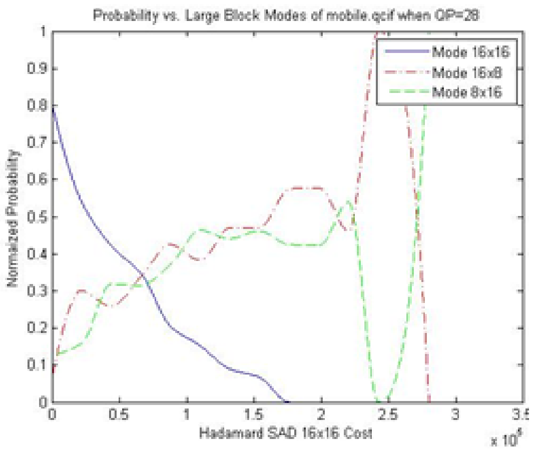
(d)



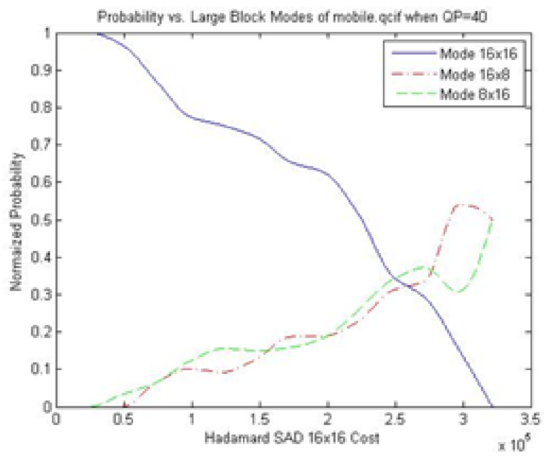
(b)



(e)



(c)



(f)

Fig. 7.3 Probability Characteristic Curve of Large Modes for Mobile QCIF Sequence at QP = 20, 24, 28, 32, 36 & 40.

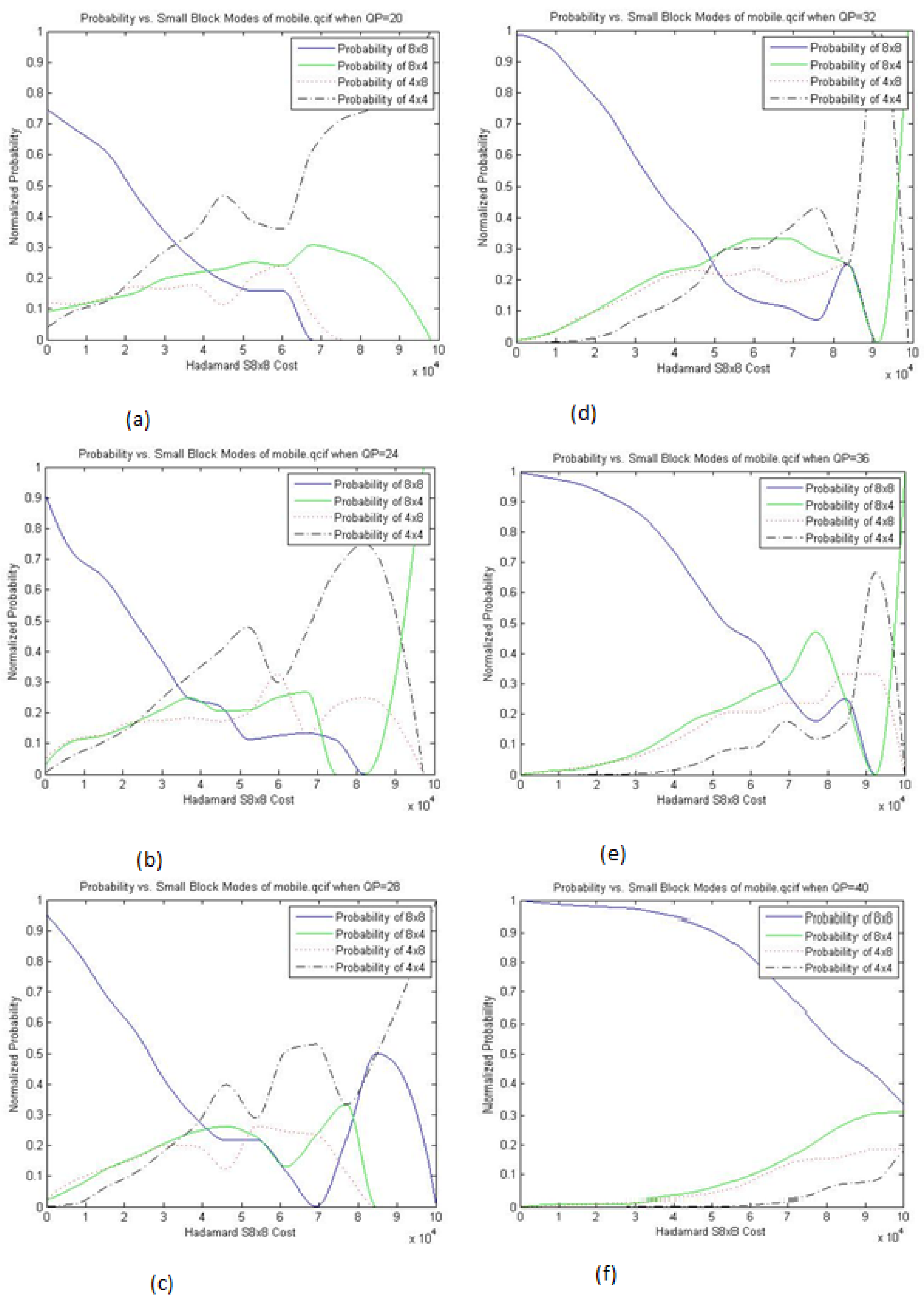


Fig. 7.4 Probability Characteristic Curve of Small Modes for Mobile QCIF Sequence at QP = 20, 24, 28, 32, 36 & 40.

Table 7.1 Thresholds T_2, T_3, T_5, T_6, and T_7 found by observation from the simulated probability versus 16×16 costs

Sequence		Large Blocks (Main Modes)		Small Blocks (Sub Modes)		
	QP	T_2	T_3	T_5	T_6	T_7
News	20	170300	20910	59400	14430	21540
	24	181500	52260	73690	21910	54710
	28	155900	62690	94940	26640	79920
	32	164100	112200	68650	39430	NAN
	36	187200	268300	114900	66410	NAN
	40	247500	267800	97100	71235	NAN
	QP	T_2	T_3	T_5	T_6	T_7
Foreman	20	122800	23350	36450	56740	16780
	24	147000	23220	39240	60100	23140
	28	151500	57860	48000	60800	37260
	32	148500	94910	58630	43580	NAN
	36	162400	191900	82630	72140	NAN
	40	231000	218500	79000	NAN	NAN
	QP	T_2	T_3	T_5	T_6	T_7
Mobile	20	148900	39690	60640	40320	32890
	24	194300	51060	91550	51320	33990
	28	212400	67450	39330	62330	41150
	32	164300	107200	48950	73110	83440
	36	201900	164900	67540	82680	87480
	40	277200	249700	101900	106500	NAN

Table 7.2 Optimized thresholds T_2, T_3, T_5, T_6, and T_7 found using polynomial curvature theorem on the data

Sequence		Large Blocks (Main Modes)		Small Blocks (Sub Modes)		
	QP	T_2	T_3	T_5	T_6	T_7
News	20	170320	23320	64200	14372	21550
	24	181151	55551	81637	21776	54722
	28	154735	66892	107351	26384	79934
	32	161443	117202	87152	38990	NAN
	36	182092	273798	141461	65703	NAN
	40	238640	273240	108200	96021	NAN
	QP	T_2	T_3	T_5	T_6	T_7
Foreman	20	122784	24600	38680	56820	16780
	24	146980	25189	42422	60274	23140
	28	151475	60920	52330	61121	37260
	32	148470	99543	64313	44120	NAN
	36	162365	169207	89880	72992	NAN
	40	230960	228200	88040	NAN	NAN
	QP	T_2	T_3	T_5	T_6	T_7
Mobile	20	146000	39084	55520	40152	33700
	24	191714	50373	82839	50981	35041
	28	210708	66709	25576	61740	42468
	32	164257	106443	28392	72172	85050
	36	204472	164174	38073	81279	89406
	40	283600	249068	61040	104504	NAN

blocks, therefore, two thresholds T_2 and T_3 has been introduced to distinguish among them. Block size 16×8 is considered as the best mode if the initial cost is greater than T_2. If the initial cost is greater than T_3 then 8×16 mode is selected. Otherwise, 16×16 mode is considered as the best mode. Similarly, as there are four small modes, three thresholds T_5, T_6, T_7 are introduced to distinguish among them. For a greater value than T_5, T_6, T_7, 8×4, 4×8 and 4×4 modes are considered as the best mode respectively. Otherwise, 8×8 mode is considered as the best sub mode. The threshold values for each mode are taken very carefully for different quantization parameters, QP = 20, 24, 28, 32, 36 & 40. Table 7.1 has shows the thresholds for different video sequences for different quantization parameters obtained by observing and analyzing the mode vs. cost characteristic curves. To find out the optimized thresholds, polynomial curvature fitting theorem has been applied. Thus, optimized thresholds values of Foreman, News and Mobile QCIF video sequence for each quantization parameter are obtained which are shown in Table 7.2. Therefore, it is possible to use the data from Table 7.2 to get the thresholds values for determining the best inter mode.

7.4.5 Proposed Algorithm with Flowchart

The probability based fast inter mode algorithm can be represented with following steps:

- Step 1: Initialize the thresholds T_2, T_3, T_5, T_6 and T_7 for a specific QP value.
- Step 2: Select a macroblock and determine its Hadamard $SAD_{16\times16}$ cost.
- Step 3: Check if the MB is suitable for SKIP mode, if not go to Step 4, if yes select skip mode as the best mode.
- Step 4: Divide the macroblock into four 8×8 sub blocks and determine Hadamard $SAD_{s8\times8}$ cost for each block and calculate them to find Hadamard $SAD_{p8\times8}$ cost.
- Step 5: If Hadamard $SAD_{16\times16}$ cost is less or equal to Hadamard $SAD_{p8\times8}$ cost choose large modes category, otherwise small modes category.
- Step 6: If Large mode category is chosen, compare the Hadamard $SAD_{16\times16}$ cost with the T_2 and T_3 to determine the best Large mode $(16 \times 16$, 16×8 and $8 \times 16)$.
- Step 7: If small mode category is chosen, compare the $SAD_{s8\times8}$ of a sub block with the T_5, T_6 and T_7 to determine the best small mode (8×8, 8×4, 4×8 and 4×4).

The flow chart of the proposed algorithm is shown in Fig.7.5. Based on this algorithm it is possible to determine the best inter mode without calculating the rate distortion cost function.

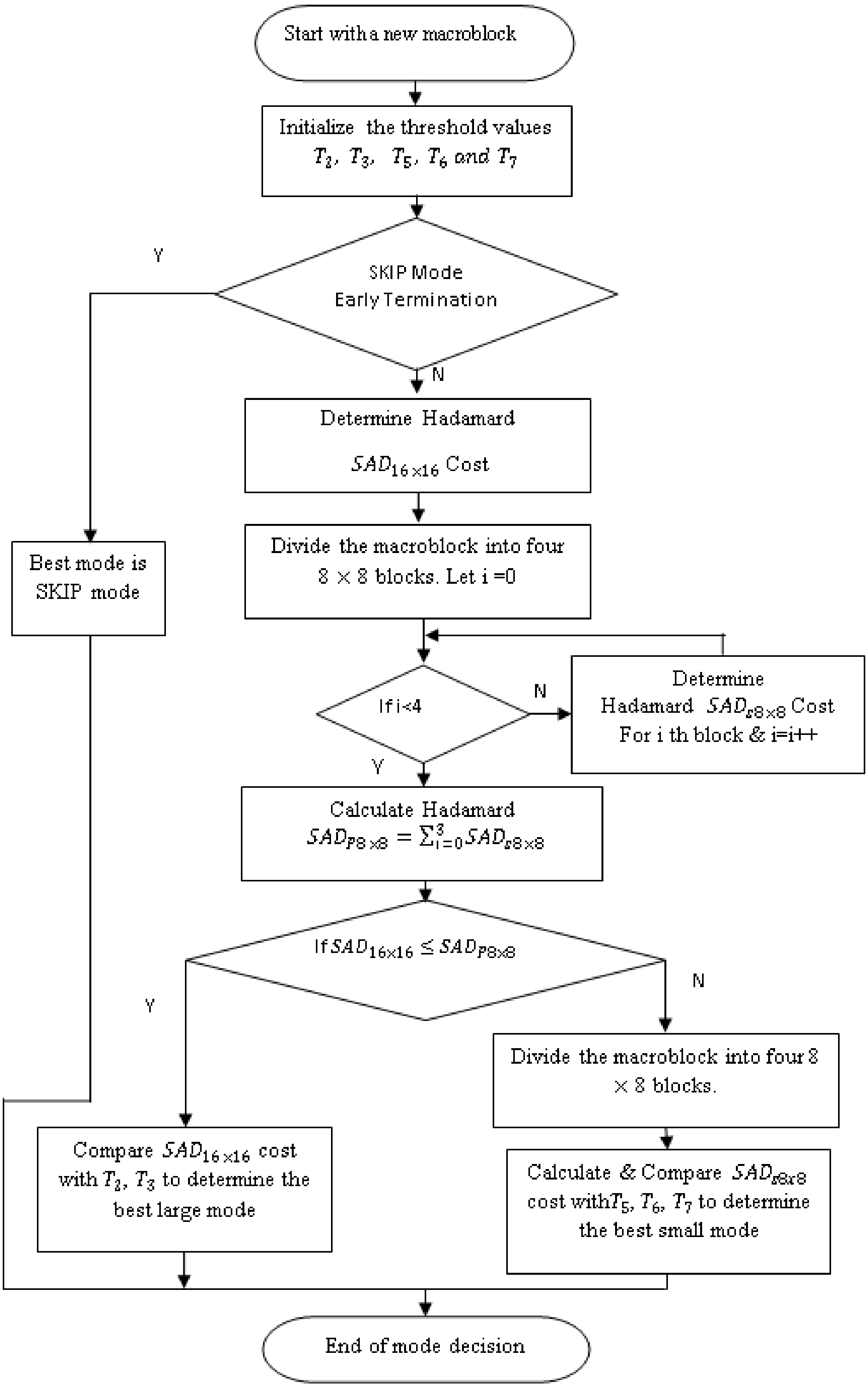

Start with a new macroblock
Initialize the threshold values T_2, T_3, T_5, T_6 and T_7
SKIP Mode Early Termination
Y
N
Best mode is SKIP mode
Determine Hadamard $SAD_{16\times16}$ Cost
Divide the macroblock into four 8 × 8 blocks. Let i =0
If i<4
N
Y
Determine Hadamard $SAD_{s8\times8}$ Cost For i th block & i=i++
Calculate Hadamard $SAD_{P8\times8} = \sum_{i=0}^{3} SAD_{s8\times8}$
If $SAD_{16\times16} \leq SAD_{P8\times8}$
Y
N
Divide the macroblock into four 8 × 8 blocks.
Compare $SAD_{16\times16}$ cost with T_2, T_3 to determine the best large mode
Calculate & Compare SAD_{s8x8} cost with T_5, T_6, T_7 to determine the best small mode
End of mode decision

Fig. 7.5 Flow diagram of proposed Probability based Fast Inter Mode decision algorithm.

Table 7.3 RD performance and encoding time reduction of probability based inter mode decision algorithm

Sequence		Performance data					
Foreman	QP	20	24	28	32	36	40
	BIT Rate %	-8.722	-9.57	-8.50	-5.7	-2.4	-0.31
	PSNR-db	0.057	0.013	0.028	0.08	0.13	0.17
	Encoding Time (%)	63.70	58.62	11.28	39.7	40.0	43.66
News	QP	20	24	28	32	36	40
	BIT Rate %	-19.92	-21.46	-22.3	-6.0	-4.8	-3.58
	PSNR-db	0.218	0.175	0.175	0.09	0.10	0.071
	Encoding Time (%)	51.717	78.50	82.57	77.9	74.1	69.53
Mobile	QP	20	24	28	32	36	40
	BIT Rate %	-2.6087	-3.85	5.71	-6.7	-5.7	-3.69
	PSNR-db	0.102	0.11	0.068	0.05	0.09	0.078
	Encoding Time(%)	58.340	49.30	48.91	53.3	54.9	59.15

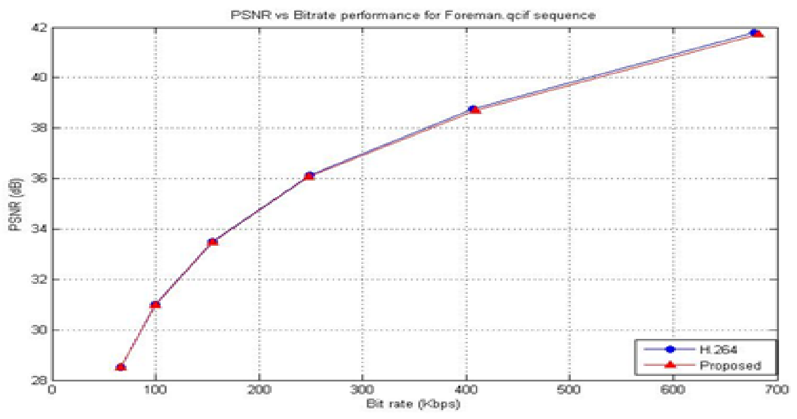
7.5 Results and Discussion

Proposed probability based mode decision algorithm was tested using the first 100 frames of different video sequences of QCIF format. Among them “Foreman” has medium motion changes with dominant luminance changes. “News” has a low spatial details and changes in motion. “Mobile” has a complex horizontal and vertical motion with slow zooming and panning. The experiment is carried out in the JVT 18.2 [13] encoder and the parameters are listed as below:

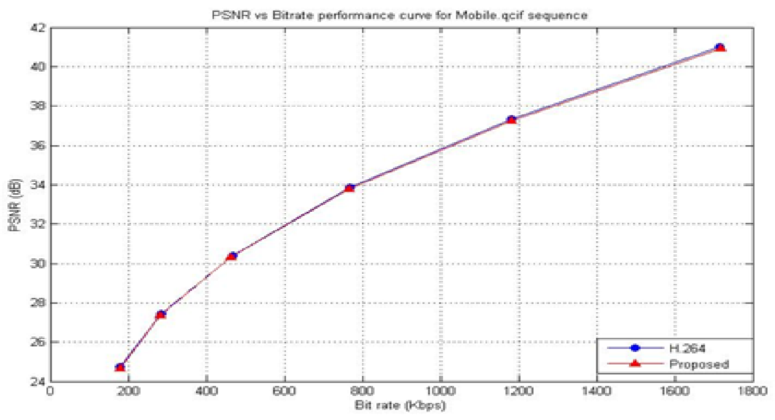
- CABAC enabled;
- GOP structure is IPPPP;
- Maximum search range for motion estimation is 32;
- QP values are 20, 24, 28, 32, 36 and 40.

The thresholds determined by the experiment varied very slightly for different video sequences. Compared to the original H.264/ AVC video encoder, proposed algorithm has reduced the computational time significantly by around 60% to 65% for different QP values which are shown in Table 7.3.

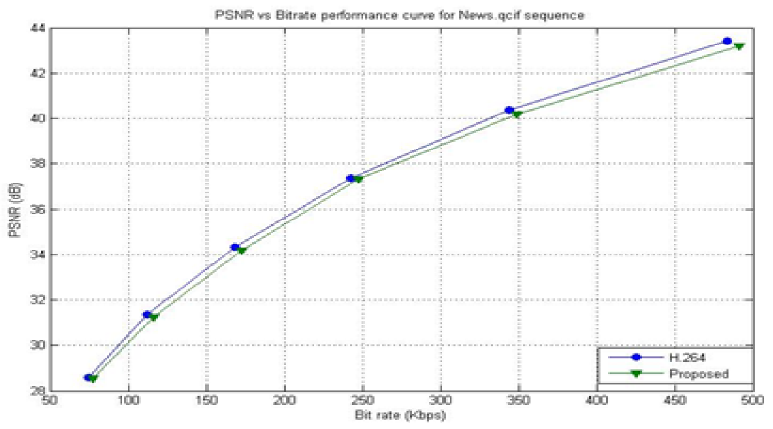
Though the bit rate increases and PSNR has reduced slightly, but the degradation of video quality for is very low and considered almost negligible. The PSNR and bit rate performance are shown in Fig. 7.6. Also the RD performance and encoding time reduction



(a)

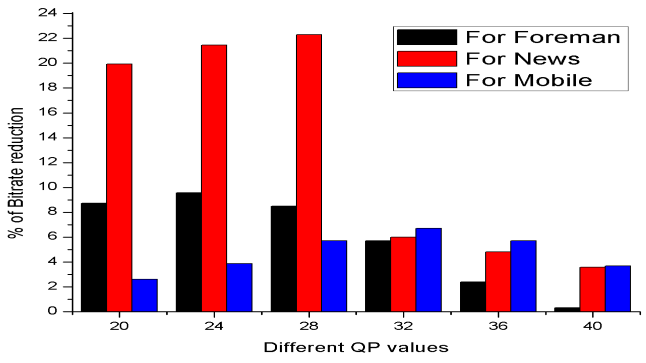


(b)

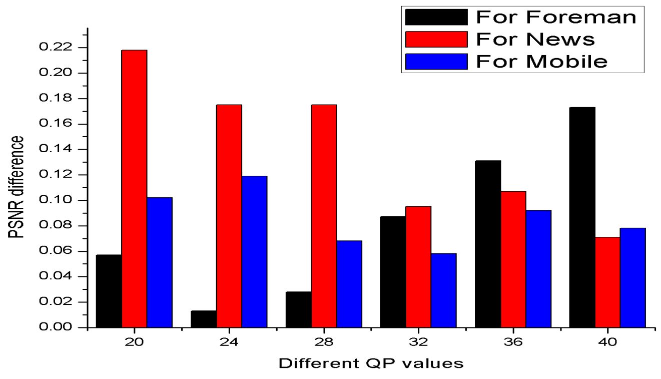


(c)

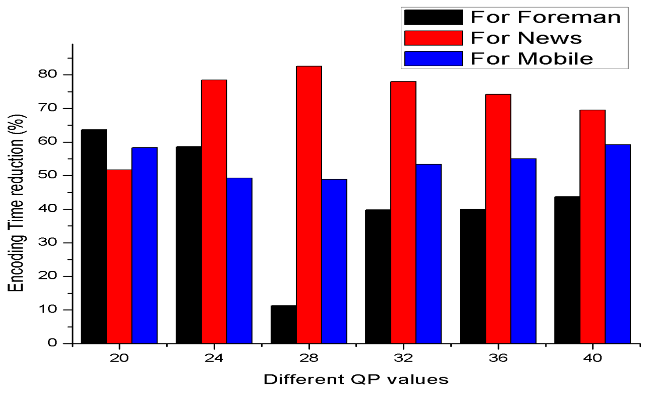
Fig. 7.6 PSNR vs. Bit Rate performance curve for (a) Foreman QCIF, (b) Mobile QCIF, (c) News QCIF Video Sequence.



(a)



(b)



(c)

Fig. 7.7 RD performance and encoding time reduction of probability based inter mode decision algorithm (a) Percentage of bit rate reduction, (b) PSNR difference, (c) Encoding time reduction.

of probability based fast inter mode decision algorithm is shown in Fig. 7.7. Finally in Fig. 7.8, the reconstructed video frames for Foreman QCIF video sequence using proposed

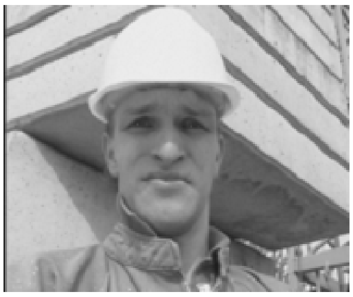
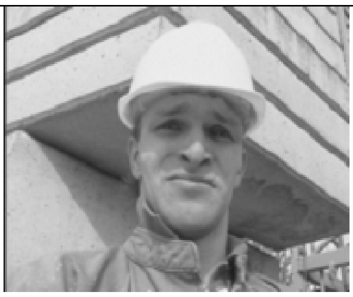
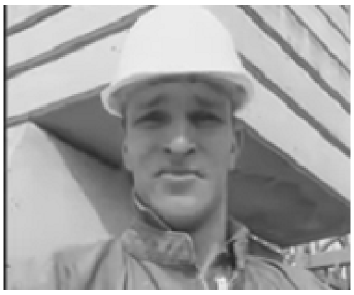
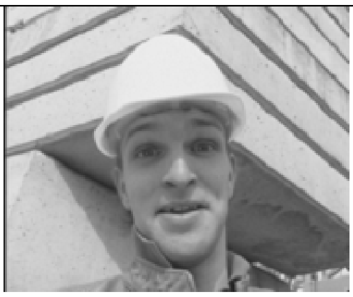
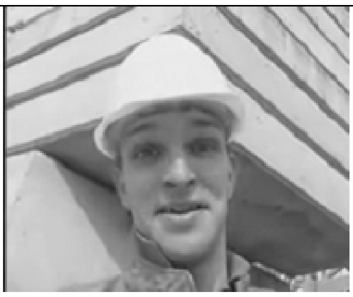
Frame No	Before Reconstruction	After Reconstruction
20		
40		
60		
80		

Fig. 7.8 Comparison of reconstructed video sequence with original one for proposed algorithm

algorithm is shown for observation. Therefore, the proposed algorithm is applicable for real time applications where reducing computational time is preferable compensating the video quality with negligible distortion.

Chapter 8

PERFORMANCE ANALYSIS OF MOTION ESTIMATION ALGORITHMS IN VIDEO CODING

In this chapter, the performance of commonly used Motion Estimation (ME) algorithms are analyzed for hardware implementation. Since number of searching point is not a direct measure of hardware performance, we first define a cost function of accessing data in memory. Then, we make adjustments on the existing ME algorithms to minimize the problem of memory access by reordering the sequence of searching points, where the number of searching points and MAE/PSNR remains the same as the original algorithms. Finally, we calculate the memory cost improvement of each algorithm. Using the present analysis we see that the cost function of accessing data in memory can be saved from 68% to 80% and the number of search point can be saved from 87% to 94%. The experimental results showed that the ranking of memory access cost for small motion video is $SCDS > BBGDS > CDS > DS > 4SS > 3SS > FS$ and for high motion video the ranking of memory access cost is $BBGDS > CDS > SCDS > 4SS > DS > 3SS > FS$ with lower computational complexity and similar quality is maintained.

8.1 Challenges and Literature Survey

Motion estimation (ME) has proven to be effective to exploit the temporal redundancy of video sequences and is therefore a central part of the ISO/IEC MPEG-1, MPEG-2, MPEG-4 and CCITT H.261 / ITU-TH.263 and H.264 video compression encoder algorithms [78]. Motion estimation [118] is used also for other applications than video encoding, like image stabilization, motion segmentation, and image analysis. However, the requirements for these applications differ significantly for video encoding algorithms: Motion vectors [119] have to reflect the real motion within the image sequence; otherwise the algorithms will not give the desired results. For video encoding the situation is different. Motion vectors are used to compensate the motion within the video sequence and only the remaining signal (prediction error) has to be encoded and transmitted. Therefore, the motion vectors have to be selected in order to minimize the prediction error and the number of bits required to code the prediction error. Motion estimation is in most cases based on a search scheme which tries to find the best matching position of a 16×16 macro block (MB) of the current frame with a 16×16 blocks within a predetermined or adaptive search range in the previous frame. The matching position relative to the original position is described as the motion vector which is (after subtraction of the predictor and variable length coding) transmitted in the bit stream to the video decoder.

Fast ME algorithms [120] shown that they can save a lot of searching points, but they all based on an assumption that all data are accessed randomly, that means they assume the whole memory block is refreshed for each searching points and also the sequence of searching points does not accounted under this assumption. It is very different from the actual hardware environment that directs implementation of these algorithms may not be efficient. There are two major factors for a ME algorithm affects the hardware to be efficient. One is memory access efficiency and another is pipeline efficiency [121]. For memory access, while using a simple architecture that the video block data is accessed row-by-row (or column-by-column); if the block distortions are measured on neighboring locations, it is not necessary to refresh the whole block thus save the bandwidth of the data bus. For instant, one step up, down, left, right movements are very efficient since it only requires replacing single row or column of memory instead of reloading the whole block. We can see that the memory access will be efficient if we keep the searching steps small. On the contrary, we should prevent diagonal move or random jump since these moves requires refreshing more data or almost the whole memory block.

For pipeline efficiency, if a fixed searching sequence can be defined, all video data can be queued up so the block distortion measure can keep working thus the hardware can be utilized. But for fast ME algorithms the further searching path always depends on distortions of the previous searched points [122]. For handling uncertain future searching point, prediction can be used to pre-fetch the data for next searching point. But when the prediction is wrong the penalty might be serious since the pipeline need to be cleared and data should be reloaded. If we want a fast ME algorithm to be pipeline friendly we should keep the number of branches to be as small as possible or make the branches to be predictable. In [123] a low-power parallel tree architecture is proposed for full search block-matching motion estimation as parallel tree architecture exploits the spatial data correlations between parallel candidate block searches for data sharing, which effectively eliminates huge amount of data access bandwidth while consumes fewer hardware resources compared with array-based architectures. Besides, this architecture can also eliminate redundant computation without pipeline latency and excess power consumption caused by register shifting and redundant memory accessing in array-based architectures. In [124] a memory hierarchy model for a full search motion estimation core. The motion estimation is the most complex module in a video encoder requiring intensive computation and high memory bandwidth, mainly when the focus is high definition videos. The proposed memory hierarchy model is based on a data reuse scheme considering the full search algorithm features and expressively reduces the external memory bandwidth required for the motion estimation process and it provides a very high data throughput for the ME core.

In [125] systolic array architecture for FSBMA is introduced by RTL-level VHDL for using as a motion estimation unit in low bit rate and real-time applications such as video telephony. The three-step search (TSS) algorithm has been widely used as the motion estimation technique in some low bit-rate video compression applications, owing to its simplicity and effectiveness. However, TSS uses a uniformly allocated checking point pattern in its first step, which becomes inefficient for the estimation of small motions. A new three-step search (NTSS) algorithm is introduced in [126], employing a center-biased checking point pattern in the first step, which is derived by making the search adaptive to the motion vector distribution, and a halfway-stop technique to reduce the computation cost. Based on the real world image sequence's characteristic of center-biased motion vector distribution, a new four-step search (4SS) algorithm with center-biased checking point pattern for fast block motion estimation is proposed in [127]. The proposed 4SS performs better than the well-known three-step search and has similar performance to the new three-step search (N3SS) in terms of motion compensation errors. A block-based gradient descent search (BBGDS) algorithm is proposed in [128] to perform block motion estimation in video coding. The minimum within the checking block is found, and the gradient descent direction where the minimum is expected to lie is used to determine the search direction and the position of the new checking block. In block motion estimation, search patterns with different shapes or sizes and the center-biased characteristics of motion-vector distribution have a large impact on the searching speed and quality of performance an algorithm using a cross-search pattern as the initial step and large/small diamond search (DS) patterns as the subsequent steps for fast block motion estimation is introduced in [129] and they also that CDS is much more robust, and provides faster searching speed and smaller distortions than other popular fast block-matching algorithms. In [130] two cross-diamond-hexagonal search (CDHS) algorithms, which differ from each other by their sizes of hexagonal search patterns is proposed. These algorithms basically employ two cross-shaped search patterns consecutively in the very beginning steps and switch using diamond-shaped patterns. To further reduce the checking points, two pairs of hexagonal search patterns are proposed in conjunction with candidates found located at diamond corners. The widespread uses of block-based inter frame motion estimation for video sequence compression in both MPEG and H.263 a standard is due to its effectiveness and simplicity of implementation. Nevertheless, the high computational complexity of the full-search algorithm has motivated a host of suboptimal but faster search strategies.

A popular example is the three-step search (TSS) algorithm. However, its uniformly spaced search pattern is not well matched to most real-world video sequences in which the motion vector distribution is non-uniformly biased toward the zero vector. Such an observation inspired the new three-step search (NTSS) which has a center-biased search pattern and supports a halfway-stop technique. It is faster on average, and gives better motion estimation as compared to the well-known TSS. Later, the four-step search (4SS) algorithm was introduced to reduce the average case from 21 to 19 search points, while maintaining a performance similar to NTSS in terms of motion compensation errors. An unrestricted center-biased diamond search (UCBDS) algorithm in [131] is introduced which is more efficient, and robust over a wide range of test video sequences. Inspired from the above, in this chapter an efficient, effective and robust approach is proposed to reduce the numbers of search in motion estimation. Besides the performance are analyzed and the ranking of the commonly used motion estimation algorithm is proposed for low and high motion video content.

8.2 Memory Access Cost Function

The definition of memory cost function is based on the following assumptions for simple hardware architecture [110]:

1. Time of accessing row or column memory is the same.
2. No carry-in from previous motion estimation.
3. Single port memory is used.

4. The video is on a 2D array in the memory.

The memory cost function is defined as:

$$C_{mem}(x,y) = min(M,N) + \sum_{i=2}^{n} C_{mem}(sp_i, sp_{i-1s}) \tag{8.1}$$

$$C_{mem}(sp_i, sp_j) = \begin{cases} D_{L1}(sp_i, sp_j) + \lambda_1.4K & \text{while } d_{L1(sp_i,sp_j)<min(M,N)}; \\ min(M,N) & \text{else.} \end{cases} \tag{8.2}$$

$$d_{L1}(sp_i, sp_j) = |x_i - x_j| + |y_i - y_j| \tag{8.3}$$

where, C_{mem} is the memory cost function for motion estimation at point (x,y), M ,N is the macro block size, usually 16×16, C_{mem} is the memory cost for moving from searching point sp_j to sp_i , d_{L1} is the City Block distance between the searching point sp_j and sp_i, City Block distance is used since it is the same as the number of row/columns need to be replaced in the movement, n is the number of searching points.

8.3 Adjustment for Memory Access Friendly on ME Algorithm

8.3.1 Full Search

Firstly, it is the exhaustive search that scans all the points in the searching area and it is trivial to form a pattern for efficient memory access. For a $(\pm 7, \pm 7)$ area, there are 225 searching points and the memory cost will be $16 + 224 = 240$. An example of full searching is shown in Fig.8.1.

8.3.2 Three Step Search (3SS)

3SS uses square search patterns three times with the searching pattern size reduced by half for each step. For a searching area $(\pm 7, \pm 7)$, searching patterns are 7×7, 5×5 and 3×3. The number of searching point is fixed at 25. The memory friendly algorithm is straight forward and it is adjusted as the followings:

- Step 1: Do the ordinary first step start corner and stop and center clockwise and a temporary minimum point is found.

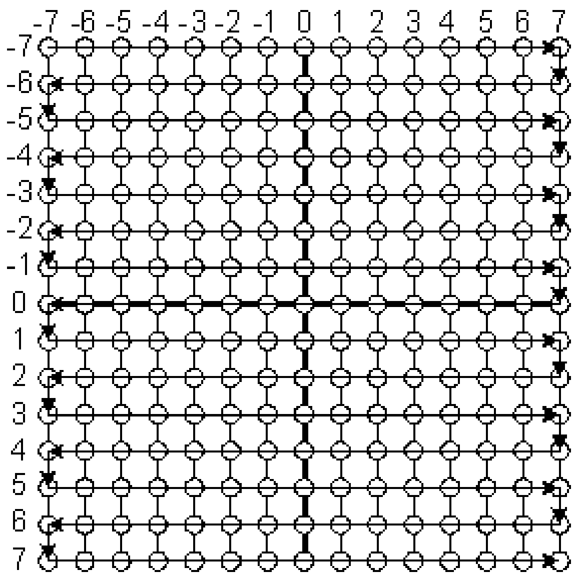


Fig. 8.1 Full search

- Step 2: Set the search pattern center at the minimum point found in step 1 and set the starting point that closest to the end point in step 1 (center). Then go through the searching pattern.

- Step 3: Set the search pattern center at the minimum point found in step 2 and set the starting point that closest to the end point in step 2. Then go through the searching pattern.

Examples of 3SS searching with global minimum point at (0, 0) and (-7,-7) are shown in Fig. 8.2 and Fig. 8.3. The search point (Spt) and corresponding memory cost (C_{mem}) is given in Table 8.1 and Table 8.2.

Table 8.1 Search point and C_{mem} of 3SS searching with global minimum at (0, 0)

Spt	1	2	3	4	5	6	7	8	9	10	Total
C_{mem}	16	4	4	4	4	4	4	4	4	2	73
Spt	11	12	13	14	15	16	17	18	19	20	
C_{mem}	2	2	2	2	2	2	2	2	1	1	
Spt	21	22	23	24	25						
C_{mem}	1	1	1	1	1						

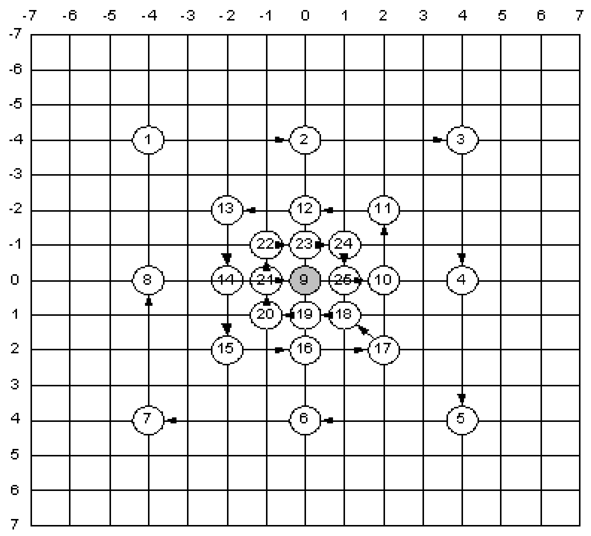


Fig. 8.2 3SS searching with global minimum point at (0, 0)

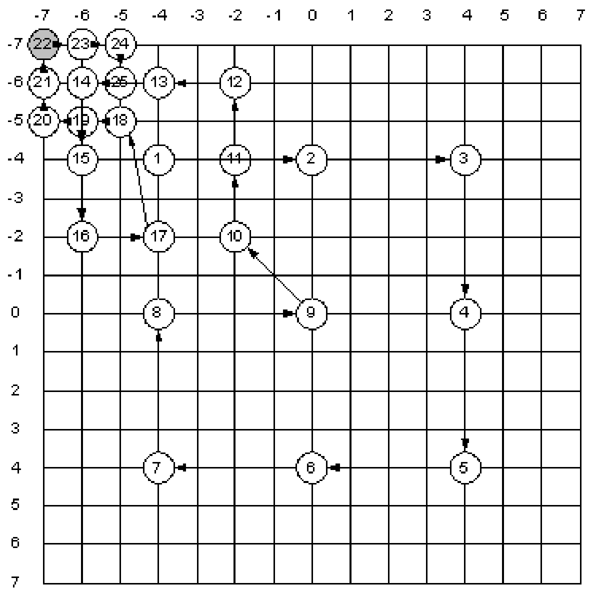


Fig. 8.3 3SS searching with global minimum at (-7,-7)

Table 8.2 Search point and C_{mem} of 3SS searching with global minimum at (-7, -7)

Spt	1	2	3	4	5	6	7	8	9	10	Total
C_{mem}	16	4	4	4	4	4	4	4	4	4	
Spt	11	12	13	14	15	16	17	18	19	20	
C_{mem}	2	2	2	2	2	2	2	4	1	1	73
Spt	21	22	23	24	25						
C_{mem}	1	1	1	1	1						

8.3.3 Four Step Search (4SS)

4SS also uses square search patterns, but with four times and different search pattern size. For a searching area $(\pm 7, \pm 7)$, searching patterns for first three steps is 5×5 and the final step is 3×3. The number of searching point is varying between 17 and 27. The memory friendly algorithm is adjusted number of searching point is varying between 17 and 27. The memory friendly algorithm is adjusted as the followings:

- Do the ordinary first step start corner and stop and center clockwise and a temporary minimum point is found.
- Set the search pattern center at the minimum point found in step 1. Then find the starting point and go through the unsearched points anti-clockwise. To find the starting point, find an unsearched point that the previous point is searched.
- Set the search pattern center at the minimum point found in step 2. Then find the starting point and go through the unsearched points clockwise. To find the starting point, find an unsearched point that the previous point is searched.
- Set the search pattern center at the minimum point found in step 3 and set the starting point that closest to the end point in step 3. Then go through the searching pattern.

Examples of 4SS searching with global minimum point at (0,0) and (7,-7) are shown in the Fig. 8.4 and Fig.8.5. The search point (Spt) and corresponding memory cost (C_{mem}) is given in Table 8.3 and Table 8.4.

8.3.4 Diamond Search (DS)/Unrestricted Center Biased Diamond (UCBDS)

DS uses a diamond shape pattern instead of square shape pattern that 3SS and 4SS does and with a converging mechanism like 4SS. The major different from 3SS and 4SS is that it does not limit the total searching step. Although the theoretical maximum number of searching points is really high, the average number of searching point is even smaller than 4SS. The memory friendly algorithm is adjusted as the followings:

- Step 1: Do the first step (Large Diamond Search Pattern, LDSP) start at top and stop and center clockwise : and a temporary minimum point is found. If minimum point is (0, 0) go to Step 3 directly.
- Step 2: Set the LDSP center at the minimum point found in step 1 (last step). Then find the starting point and go through the unsearched points reverse direction from last step. To find the starting point, find an unsearched point that the previous point is searched. If minimum point is center go to Step 3; if not, repeat the step again.
- Step 3: Set the search pattern (Small Diamond Search Pattern, SDSP) center at the minimum point found in step 2 and set the starting point that closest to the end point in step 3. Then go through the searching pattern.

Examples of DS searching with global minimum point at (0,0) and (2,-7) are shown in the Fig. 8.6 and Fig. 8.7. The search point (Spt) and corresponding memory cost (C_{mem}) is given in Table 8.5 and Table 8.6.

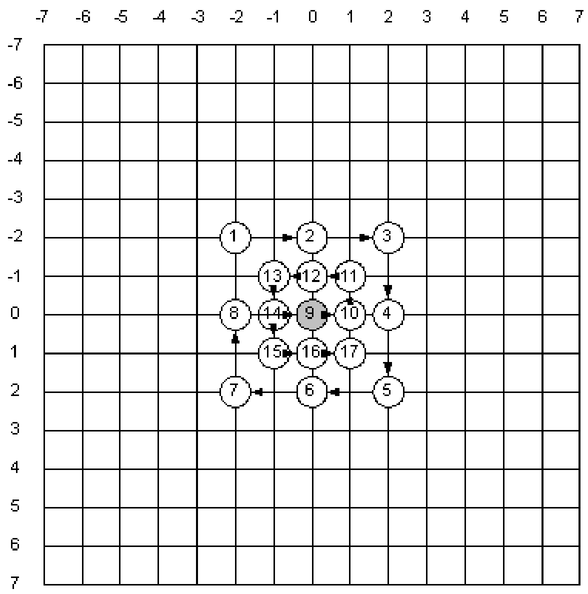


Fig. 8.4 4SS searching with global minimum point at (0, 0)

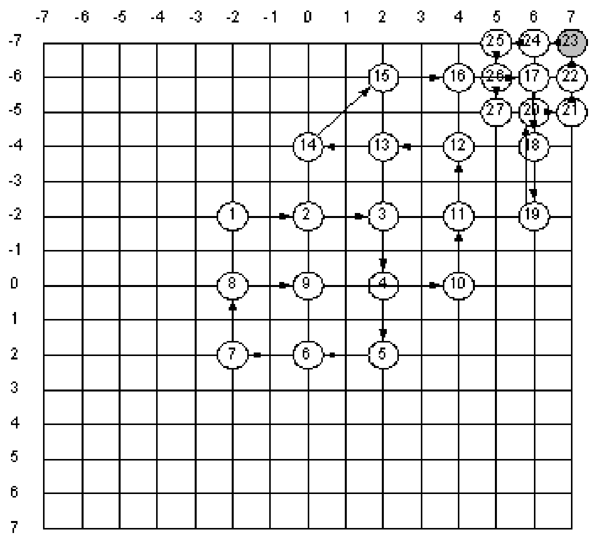


Fig. 8.5 4SS searching with global minimum point at (7,-7)

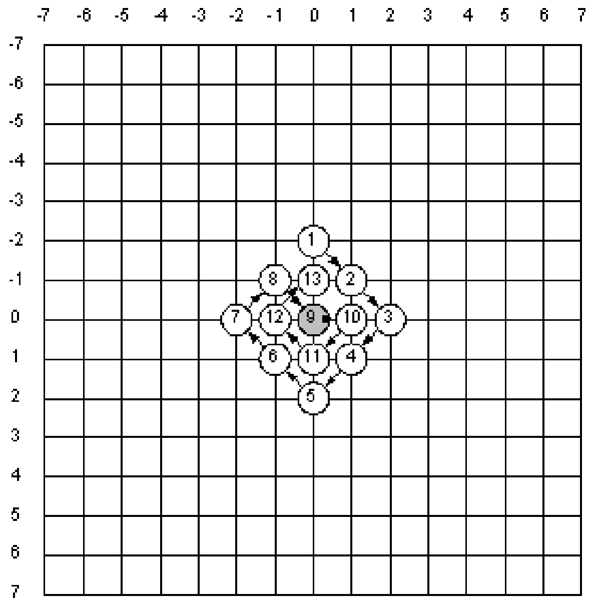


Fig. 8.6 DS searching with global minimum point at (0, 0)

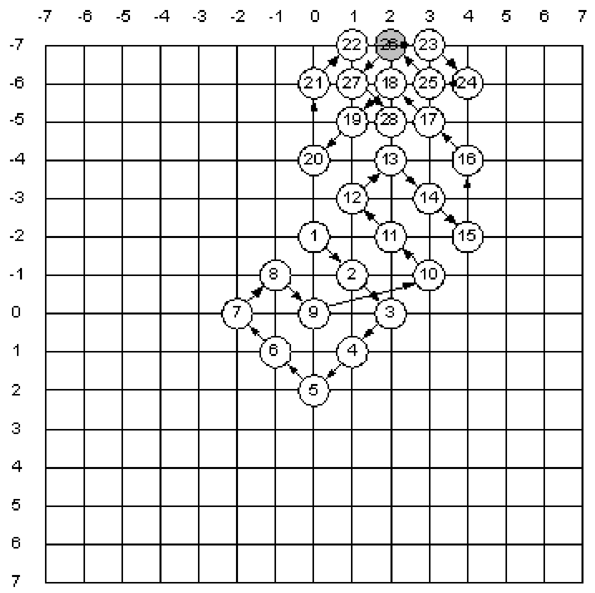


Fig. 8.7 DS searching with global minimum point at (2,-7)

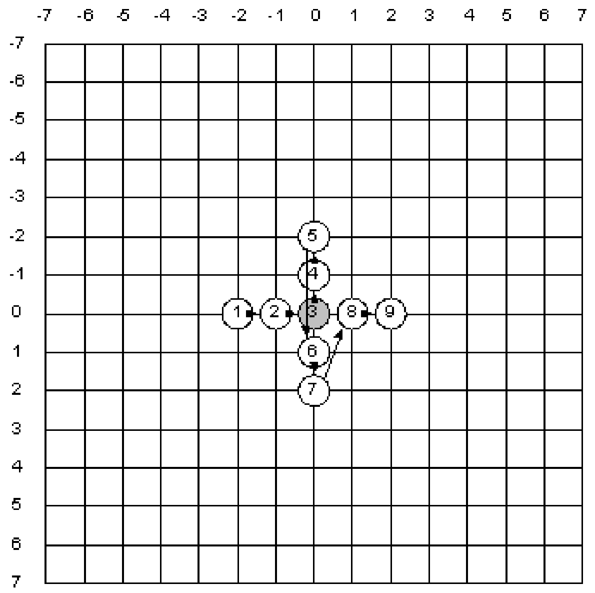


Fig. 8.8 CDS searching with global minimum point at (0, 0)

Table 8.3 Search point and C_{mem} of 4SS searching with global minimum at (0, 0)

Spt	1	2	3	4	5	6	7	8	9	10	Total
C_{mem}	16	2	2	2	2	2	2	2	2	1	40
Spt	11	12	13	14	15	16	17				
C_{mem}	1	1	1	1	1	1	1				

Table 8.4 Search point and C_{mem} of 4SS searching with global minimum at (7, -7)

Spt	1	2	3	4	5	6	7	8	9	10	Total
C_{mem}	16	2	2	2	2	2	2	2	2	4	65
Spt	11	12	13	14	15	16	17	18	19	20	
C_{mem}	2	2	2	2	4	2	2	2	3	1	
Spt	21	22	23	24	25	26	27				
C_{mem}	1	1	1	1	1	1	1				

Table 8.5 Search point and C_{mem} of DS searching with global minimum at (0, 0)

Spt	1	2	3	4	5	6	7	8	9	10	Total
C_{mem}	16	2	2	2	2	2	2	2	2	1	41
Spt	11	12	13	14							
C_{mem}	2	2	2	2							

Table 8.6 Search point and C_{mem} of DS searching with global minimum at (2, -7)

Spt	1	2	3	4	5	6	7	8	9	10	Total
C_{mem}	16	2	2	2	2	2	2	2	2	4	71
Spt	11	12	13	14	15	16	17	18	19	20	
C_{mem}	2	2	2	2	2	2	2	2	2	2	
Spt	21	22	23	24	25	26	27	28			
C_{mem}	2	2	2	2	1	2	2	2			

8.3.5 Cross Diamond Search (CDS)

CDS further improve the DS by changing the first step searching pattern. It uses a Large Cross Searching Pattern (LCSP) in the first step and introduced a new technique called first step stop that can substantially reduce the minimum number of searching points for small

Table 8.7 Search point and C_{mem} of CDS searching with global minimum at (0, 0)

Spt	1	2	3	4	5	6	7	8	9	Total
C_{mem}	16	1	1	1	1	3	1	3	1	28

motion videos. Since there is no one-way-trip for the LCSP, some memory cost is wasted. The memory friendly algorithm is adjusted as the followings:

- Step 1: Do the first step (LCSP) with the sequence shown in points 1-9 from Fig.8.8 and Table 8.7 and a temporary minimum point is found. If minimum point is (0, 0), end the process.
- Step 2: Search the half diamond points at the direction of the minimum point found in step 1 (e.g. point 10, 11 in Fig. 8.9 and Table 8.8), if the minimum point is surrounded by 4 searched points, end the process.
- Step 3: Set the search pattern (SDSP) center at the minimum point found in step 2 and set the starting point that closest to the end point in step 2. Then go through the searching pattern.

Examples of CDS searching with global minimum point at (0, 0), (2, -7) and (-3,-4) are shown in the Fig. 8.8, Fig.8.9 and Fig.8.10. The search point (Spt) and corresponding memory cost (C_{mem}) is given in Table 8.7, Table 8.8 and Table 8.9.

Table 8.8 Search point and C_{mem} of CDS searching with global minimum at (2,-7)

Spt	1	2	3	4	5	6	7	8	9	10	11	Total
C_{mem}	16	1	1	1	1	3	1	3	1	4	2	34

8.3.6 Block based Gradient Decent Search (BBGDS)

BBGDS is a very simple algorithm that only uses a 3×3 square as the search pattern. A MV will be decided if a searched point is found as minimum distortion among eight surrounding points. The memory friendly algorithm is adjusted as the followings:

- Step 1: Do the first step (3×3 square) with the sequence shown in points 1-9 from Fig.8.11 and Table 8.10 and a temporary minimum point is found. If minimum point is (0, 0), end the process.

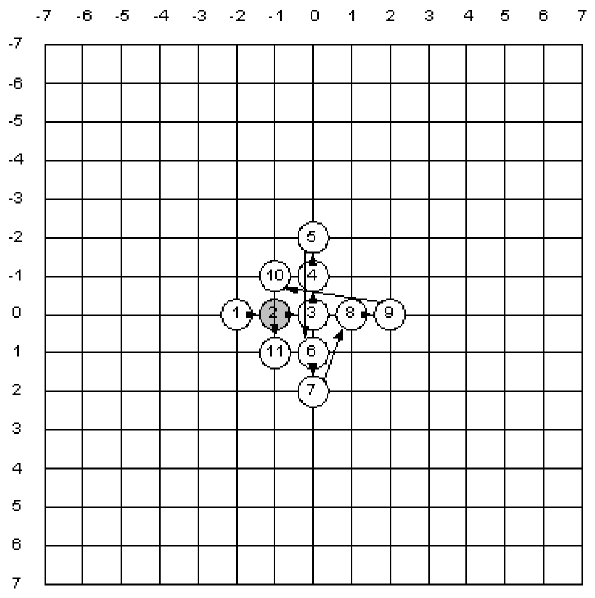


Fig. 8.9 CDS searching with global minimum point at (2,-7)

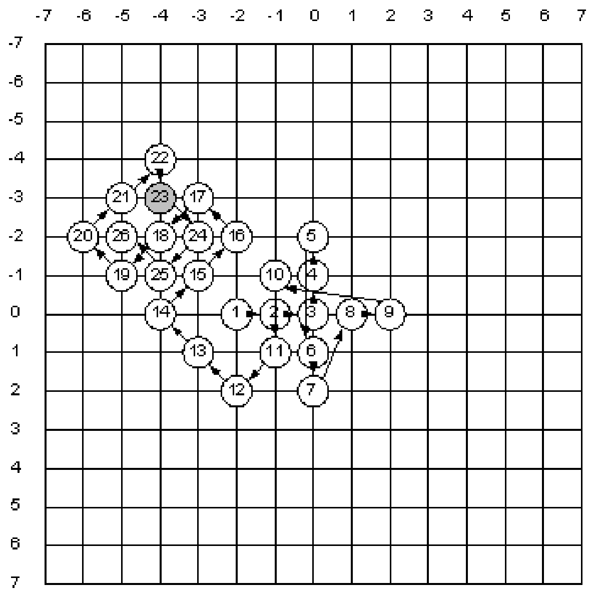


Fig. 8.10 CDS searching with global minimum point at (-3, -4)

Table 8.9 Search point and C_{mem} of CDS searching with global minimum at (-3 -4)

Spt	1	2	3	4	5	6	7	8	9	10	Total
C_{mem}	16	2	2	2	2	3	1	3	1	4	63
Spt	11	12	13	14	15	16	17	18	19	20	
C_{mem}	2	2	2	2	2	2	2	2	2	2	
Spt	21	22	23	24	25	26					
C_{mem}	2	2	1	2	2	2					

- Step 2: Search the same search pattern centered at the minimum point found in step 1 (e.g. point 10, 11, 12 in Fig.8.12 and Table 8.11), if the minimum point is the minimum among all surrounding searched points, end the process. Otherwise, repeat step 2.

Examples of BBGDS searching with global minimum point at (0, 0), (7, 0) and (7,-7) are shown in the Fig. 8.11, Fig.8.12 and Fig.8.13. The search point (Spt) and corresponding memory cost (C_{mem}) is given in Table 8.10, Table 8.11 and Table 8.12.

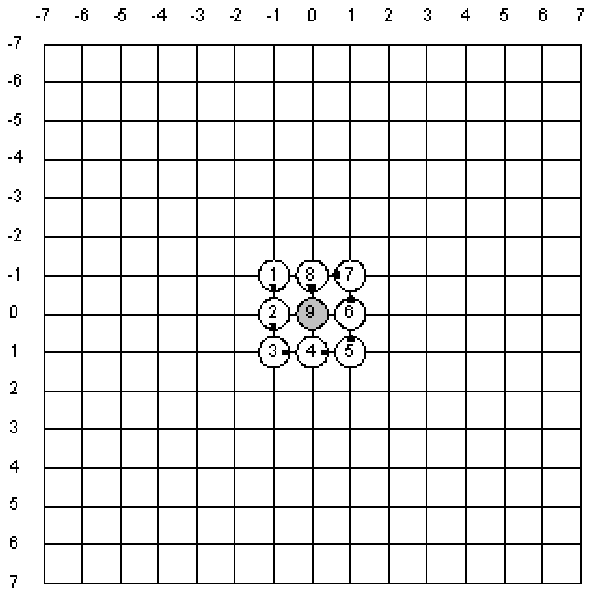


Fig. 8.11 BBGDS searching with global minimum point at (0, 0)

8.3.7 Small Cross Diamond Search (SCDS)

SCDS gives a very impressive solution for small motion video by using a Small Cross Search Pattern (SCSP) for the first step. And the following steps remain the same as CDS. Since

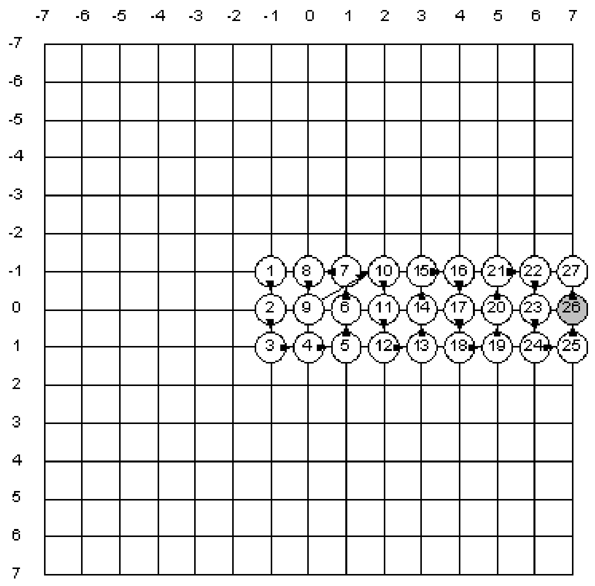


Fig. 8.12 BBGDS searching with global minimum point at (7, 0)

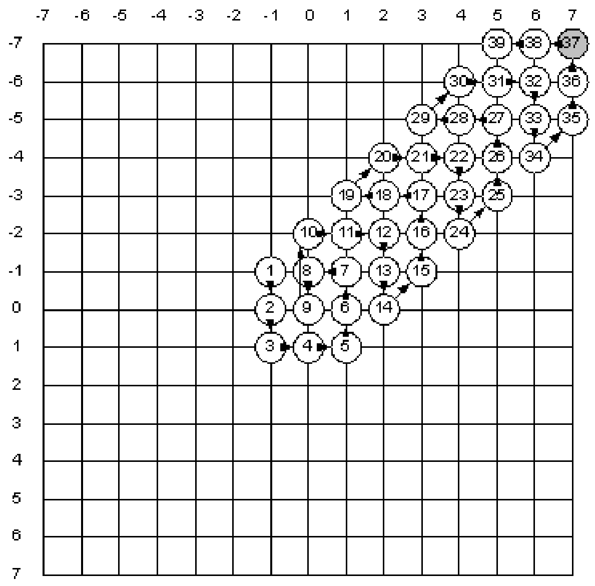


Fig. 8.13 BBGDS searching with global minimum point at (7, -7)

Table 8.10 Search point and C_{mem} of BBGDS searching with global minimum at (0, 0)

Spt	1	2	3	4	5	6	7	8	9	Total
C_{mem}	16	1	1	1	1	1	1	1	1	24

Table 8.11 Search point and C_{mem} of BBGDS searching with global minimum at (7, 0)

Spt	1	2	3	4	5	6	7	8	9	10	Total
C_{mem}	16	1	1	1	1	1	1	1	1	3	
Spt	11	12	13	14	15	16	17	18	19	20	
C_{mem}	1	1	1	1	1	1	1	1	1	1	44
Spt	21	22	23	24	25	26	27	28			
C_{mem}	1	1	1	1	1	1	1	1			

there is no one-way-trip for the SCSP, some memory access cost is wasted. The memory friendly algorithm is adjusted as the followings:

Table 8.12 Search point and C_{mem} of BBGDS searching with global minimum at (7, -7)

Spt	1	2	3	4	5	6	7	8	9	10	Total
C_{mem}	16	4	4	4	4	4	4	4	41	2	
Spt	11	12	13	14	15	16	17	18	19	20	
C_{mem}	1	1	1	1	1	1	1	1	1	2	
Spt	21	22	23	24	25	26	27	28	29	30	44
C_{mem}	1	1	1	1	1	1	1	1	1	2	
Spt	31	32	33	34	35	36	37	38	39		
C_{mem}	1	1	1	1	1	1	1	1	1		

- Step 1: Do the first step (SCSP) with the sequence shown in points 1-5 from Fig.8.14 and Table 8.13 and a temporary minimum point is found. If minimum point is (0, 0), end the process.
- Step 2: Search LCSP points and half diamond points at the direction of the minimum point found in step 1 (e.g. point 10, 11 in Fig.8.15 and Table 8.14), if the minimum point is surrounded by 4 searched points, end the process.
- Step 3: Set the search pattern (LDSP) center at the minimum point found in step 2 (last step). Then find the starting point and go through the unsearched points reverse

direction from last step (clockwise for the first time). To find the starting point, find an unsearched point that the previous point is searched. If minimum point is center go to Step 3; if not, move the center to the minimum point and repeat.

- Step 4: Set the search pattern (SDSP) center at the minimum point found in step 3 and set the starting point that closest to the end point in step 3. Then go through the searching pattern.

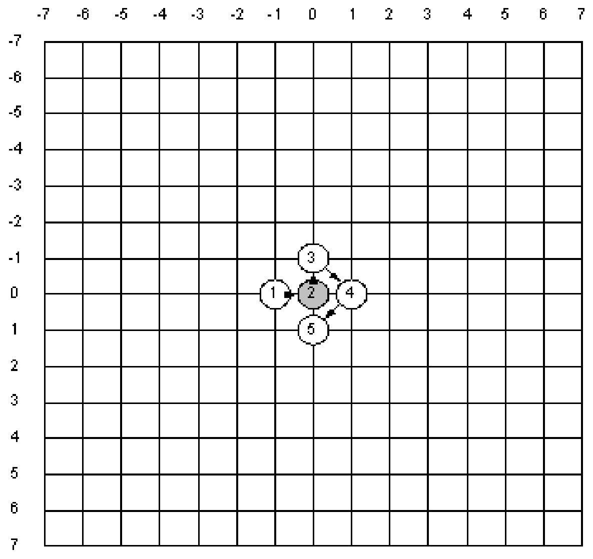


Fig. 8.14 SCDS searching with global minimum point at (0, 0)

Table 8.13 Search point and C_{mem} of SCDS searching with global minimum at (0, 0)

Spt	1	2	3	4	5	Total
C_{mem}	16	1	1	2	2	22

Examples of SCDS searching with global minimum point at (0, 0) and (-2,6) are shown in the Fig. 8.14 and Fig.8.15. The search point (Spt) and corresponding memory cost (C_{mem}) is given in Table 8.13 and Table 8.14.

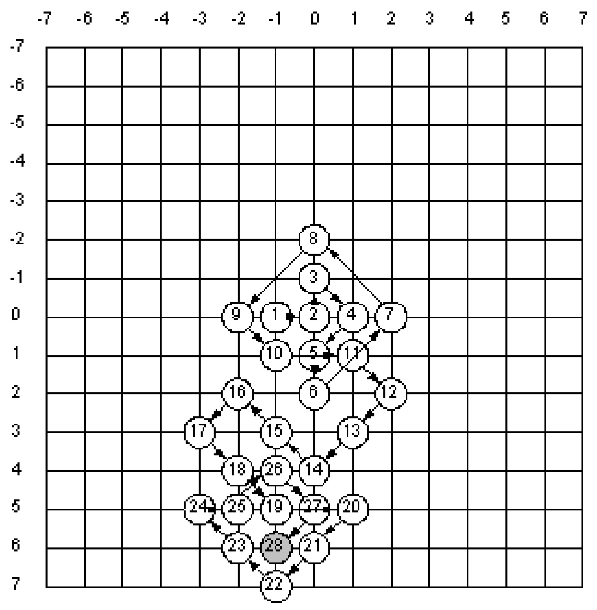


Fig. 8.15 SCDS searching with global minimum point at (-2, 6)

Table 8.14 Search point and C_{mem} of SCDS searching with global minimum at (-2, 6)

Spt	1	2	3	4	5	6	7	8	9	10	Total
C_{mem}	16	1	1	2	2	1	4	4	4	2	
Spt	11	12	13	14	15	16	17	18	19	20	
C_{mem}	2	2	4	2	2	2	2	2	2	2	72
Spt	21	22	23	24	25	26	27	28			
C_{mem}	2	2	2	2	1	2	2	2			

8.4 Experimental Setup

To obtain the performance of the proposed approach exhaustive the simulation been performed under the following condition:

- advanced prediction: on (i.e. [-2, +2] pel search for 8×8 blocks around the best 16×16 of a fast ME algorithm
- half-pel motion estimation: on
- disabled: rate control, error resilience, SADCT
- enabled: DC/AC-prediction
- deblocking filter: rectangular VO: off, arbitrarily shaped VO: on

- combined motion/shape/texture coding,
- h.264-quantization

Table 8.15 Comparison of Searching Point and Memory Access Cost for Zero Motion Vector

Search Algorithm	No of Search Point	Memory Access Cost
FS	225	240
3SS	25	73
4SS	17	40
DS	14	41
CDS	9	28
BBGDS	9	24
SCDS	5	22

8.5 Results and Comparison between Different Search Algorithm

Based on center biased property, most motion vectors will be within $(\pm1,\pm1)$, and especially for small motion videos, most of motion vectors will be zero motion. So, the number of searching points and memory access cost for minimum point at (0, 0) is a critical factor, from Table 8.15 it is found that CDS and SCDS can use smallest number of searching point and the memory access cost. The major different from 3SS and 4SS is that it does not limit the total searching step. Although the theoretical maximum number of searching points is really high, the average number of searching point is even smaller than 4SS. On the other hand CDS further improve the DS by changing the first step searching pattern. It uses a Large Cross Searching Pattern (LCSP) in the first step and introduced a new technique called first step stop that can substantially reduce the minimum number of searching points for small motion videos. The simulation were carried out using low, medium and high motion details sequence to obtain the best performance of different motion estimation algorithm in case of different types of video contents.

From Table 8.16 it is seen that the ranking of memory access cost for small motion video is $SCDS > BBGDS > CDS > DS > 4SS > 3SS > FS$. For high motion video, the ranking of memory access cost is *BBGDS* > *CDS* > *SCDS* > 4*SS* > *DS* > 3*SS* > *FS*. Searching point and corresponding memory access cost zero motion vector is shown in Fig.8.16 and also the percentage saving of search point and memory access cost for different video sequences are shown in Fig.8.17. From the graphical representation it is very clear that the performance of commonly used ME algorithms can be analyzed on hardware design issue. From the analysis, it is found that the cost function of accessing data in memory can be saved from 68% to 80% and the number of search point can be saved from 87% to 94%. So this approach can perform better where the memory savings is greatly required.

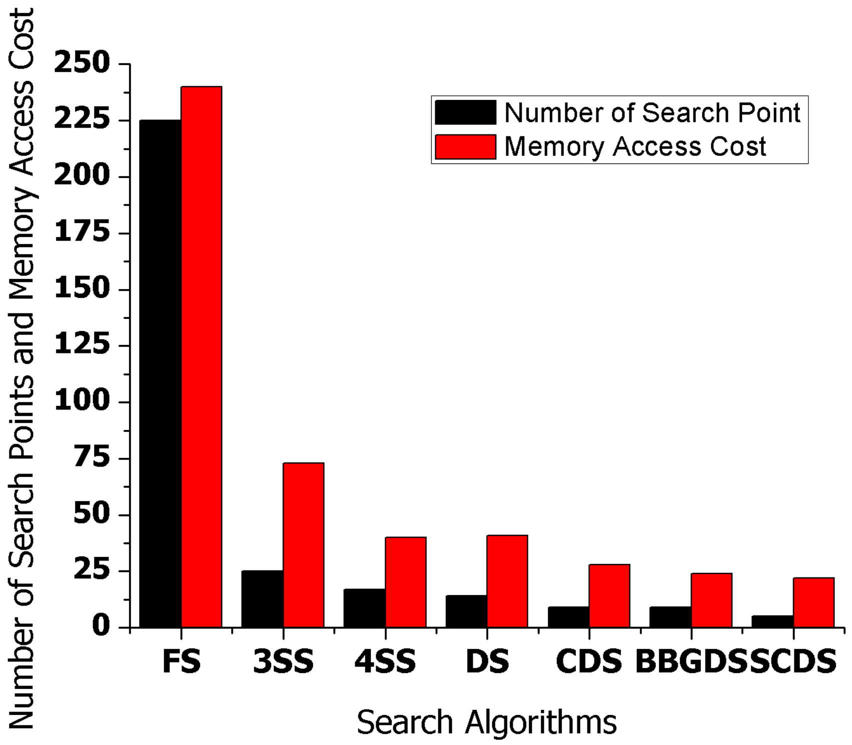


Fig. 8.16 Search point and memory access cost for zero motion vector.

8.6 Conclusion

From the analysis it is found that the cost function of accessing data in memory can be saved from 68% to 80% and the number of search point can be saved from 87% to 94%. So this approach can perform better where the memory savings is greatly needed. The experimental results also showed that the ranking of memory access cost for small motion video is SCDS > BBGDS > CDS > DS > 4SS > 3SS > FS and for high motion video the ranking of memory

Table 8.16 Performance Comparison of Motion Estimation Algorithms

(a) With Akiyo (CIF)

Akiyo (CIF)	C_{mem} /frame	MAE	MSE	PSNR	MAE/ Pixel	Spt	C_{mem} saved	Spt saved
FS	86836.0	61392.87	3.941	42.780	0.606	204.28	-	-
3SS	27475.7	61923.43	4.075	42.657	0.611	23.21	68.359%	88.64%
4SS	15300.4	61701.28	4.006	42.709	0.609	15.85	82.380%	92.24%
BBGDS	9391.5	61405.69	3.937	42.782	0.606	8.55	89.185%	95.82%
SCDS	8911.7	61497.18	3.983	42.742	0.607	5.07	89.737%	97.52%

(b) With Football (SIF)

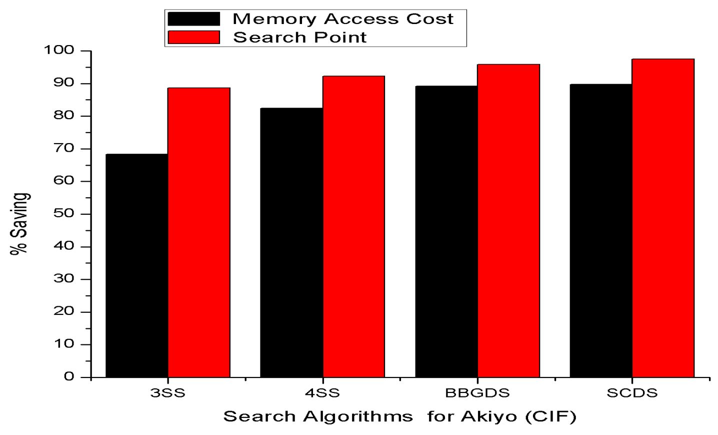
Football (SIF)	C_{mem} /frame	MAE	MSE	PSNR	MAE/ Pixel	Spt	C_{mem} saved	Spt saved
FS	71626.0	635238.94	208.086	25.620	7.520	202.05	-	-
3SS	22914.1	659199.96	230.139	25.190	7.803	23.10	68.01%	88.57%
4SS	14454.9	657698.57	227.648	25.299	7.786	17.63	79.82%	91.28%
BBGDS	9609.7	664136.70	235.656	25.206	7.862	12.87	86.58%	93.63%
SCDS	12423.0	662556.65	234.986	25.191	7.843	11.04	82.66%	94.53%

(c) With Stefan (CIF)

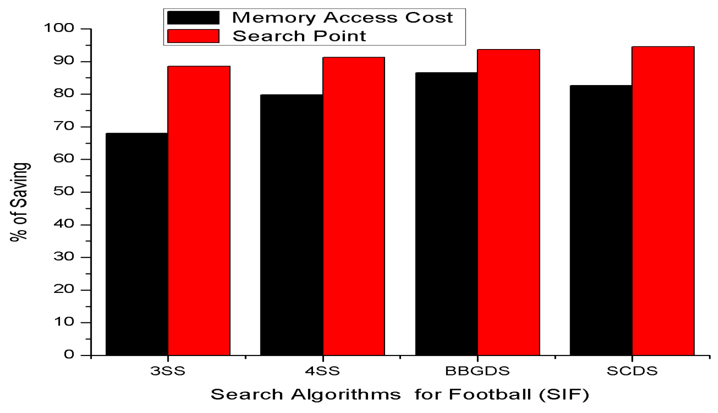
Stefan (CIF)	C_{mem} /frame	MAE	MSE	PSNR	MAE/ Pixel	Spt	C_{mem} saved	Spt saved
FS	86836.0	843357.3	303.407	24.612	8.319	204.28	-	-
3SS	27690.7	894472.87	341.119	24.191	8.823	23.30	68.11%	89.57%
4SS	18431.3	951493.35	385.155	23.672	9.386	18.65	78.77%	91.88%
BBGDS	12385.0	1003221.97	428.156	23.226	9.896	14.75	85.74%	93.81%
SCDS	18023.8	973488.53	404.538	23.469	9.603	14.33	79.24%	94.01%

access cost is BBGDS > CDS > SCDS > 4SS > DS > 3SS > FS with lower computational complexity and similar quality is maintained.From this ranking it can be concluded that if we can search using this order then we can obtained more better result for motion estimation and motion compensation.

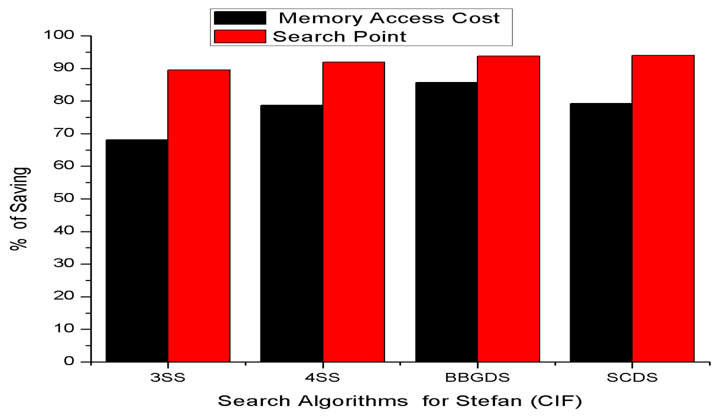
Motion estimation is the most computationally expensive task in MPEG-style video compression. Video compression is starting to be widely used in battery-powered terminals, but surprisingly little is known about the power consumption of modern motion estimation algorithms. This research describes the effort to analyze the power and performance of realistic motion estimation algorithms in both hardware and software realizations. For custom hardware realizations, this research presents a general model of VLSI motion estimation architectures. This model allows us to analyze in detail the power consumption of a large class of modern motion estimation engines that can execute the motion estimation algorithms



(a)



(b)



(c)

Fig. 8.17 Percentage of memory access cost and search point saving for (a) For Akiyo (CIF), (b) Football (SIF), (c) Stefan (CIF).

of interest to us. We compare these algorithms in terms of their power consumption and performance. For software realizations, this research provides the first detailed instruction-level simulation results on motion estimation based on a programmable CPU core. We analyzed various aspects of the selected motion estimation algorithms, such as search speed and power distribution. This work provides a guideline to two types of machine designs for motion estimation: custom ASIC (Application Specific Integrated Circuit) design and custom ASIP (Application Specific Instruction-set Processor) designs.

Chapter 9

CONCLUSION AND FUTURE WORKS

9.1 Concluding Remarks

Significant advances in digital video technology and communication sector make it possible to deliver high-quality video at low bit rates for today's networks. This state-of the-art technology enables many possible networked multimedia, which include digital television, video over the Internet, wireless and mobile video, third generation (3G) cell phones with video capabilities, and the coming generation of Internet TV (IPTV). H.264/AVC, the latest video-coding standard, offers a solution to achieve high efficiency coding. However, the performance gains of H.264/AVC come at a cost of significantly increased computational complexity. The main complexity of encoder comes from variable block size motion estimation (ME) and rate distortion optimized (RDO) mode decision method. Within video coding, mode decision plays a vital role but it is also the most computationally intensive part of encoder. In this dissertation, six efficient mode (Intra and Inter) selection schemes are proposed to reduce the computational burden of the encoder without sacrificing the quality of the video. These schemes seek to effectively utilize the available computational resources to maximize the performance of the encoder.

Firstly, a modified DC prediction (MDCP) scheme is proposed to solve the problem that DC mode used in the regions with no unified direction in H.264/AVC. The experimental result shows that a constant performance gain is obtained over H.264 with an average PSNR improvement of 0.06 dB which reaches its maximum of 0.09 dB in CIF sequences. In QCIF sequences, the average PSNR increase is 0.06 dB. Secondly, a new Z-scan scheme is proposed to further improve intra coding efficiency for the AVS standard. It is relatively easy to implement the proposed Z-scan scheme into AVS codec without changing the syntax. Experimental results demonstrate that the proposed scheme can remarkably reduce bit rates by approximately 2.1% compared with AVS codec using zigzag scan, while the PSNR of video sequences are maintained. Since improvement of intra coding is difficult to achieve, as widely known, the bitrates saving in our experiments is obviously significant especially almost without quality reduction. Besides, there is no extra computational cost in the proposed Z-scan implementation. Then, a novel rate-based macroblock classifications for fast inter and intra mode decision is proposed based on Inter 16×16 mode's residue block bit-rate. The threshold value used for the classification is also formulated in terms of QPs, which can maintain a good tradeoff between computation complex and rate-distortion performance. To further speed up the intra mode decision, a new fast Intra 4×4 mode selection algorithm is also proposed by choosing most likely modes using the low complexity SATD cost as screening function. It is demonstrated by experimental results that the proposed rate-based fast algorithm can reduce 47% to 65% of the H.264 total encoding time with negligible degradation in the rate-distortion performance. While the rate-based algorithm combined with the fast intra mode selection method could further speedup 5% to 10% of the encoding time with only little rate-distortion degradation. A relative sum of absolute difference (RSAD) based inter mode decision algorithm is proposed to determine the best inter mode based on RSAD cost instead of the R-D cost. This approach can solve the problem of SAD-based cost function in biasing to the smaller block size modes, as they are normally achieving higher prediction accuracy. Experimental results showed that the proposed RSAD-based mode decision scheme can reduce 60% to 68% of the H.264/AVC total encoding time with negligible degradation. When combined with SKIP mode, the enhanced scheme could save up to 10% more encoding time. Latter, a new probability based fast inter mode decision scheme is proposed based on the H-SAD cost instead of RD cost. The thresholds determined by the experiment varied very slightly for different video sequences. Compared to the original H.264/AVC video encoder, proposed algorithm has reduced the computational time significantly by around 60% to 78% for different QP. The performance of commonly used motion estimation algorithms are analyzed to reduce the number of search point and ranking of them. Using the present analysis, we see that the cost of accessing data in memory can be saved from 68% to 80% and the number of search point can be saved from 87% to 94%. Moreover, the experimental results showed that the ranking of memory access cost for small motion video is $SCDS > BBGDS > CDS > DS > 4SS > 3SS > FS$ and for high motion video the ranking of memory access cost is $BBGDS > CDS > SCDS > 4SS > DS > 3SS > FS$ with lower computational complexity and similar quality is maintained. The contribution made in this dissertation based on firm theoretical foundations, have minimal dependence on the obtained thresholds and are adaptive to changing sequence statistics. These can address significant practical problems that have not been properly addressed by others in previous.

In conclusion, the original contributions made for the proposed efficient mode selection schemes in this dissertation are likely to be particularly useful in implementing latest video standard compliant encoders.

9.2 Future Direction

The efficient mode selection schemes developed in this dissertation have been summarized and critically evaluated in the earlier section. This section presents some directions for further research, mainly aimed at extending the proposed schemes to achieve further flexibility with new dimension. Multi-view video receives many attentions in these years, because it can support a wide range of applications, such as 3D video communication and free viewpoint video. Inter-view predictions is also another time consuming part of multi-view video coding. The fast scheme in this dissertation can be extended to reduce the computation of inter-view prediction of multi-view video coding (MVC). The algorithms presented so far are evaluated using fixed quantized values. During the RD optimized mode decision process, the Lagrange multiplier is used as a function of the quantization parameter and therefore achieves varying degrees of performance for different video sequences. The performance may be improved for all (or for a wide variety of sequences) sequences by developing a model for estimation of the Lagrange multiplier to incorporate video sequence statistics. Adaptive number of mode selection methods described in this dissertation can be extended to improve the compression performance of conventional intra/inter-prediction method. After improvements, the developed method may be integrated in upcoming High Efficiency Video Coding (HEVC) standard [119]. High Efficiency Video Coding (HEVC) aims to substantially improve coding efficiency compared to H.264/AVC High Profile by half with comparable image quality, probably at the expense of increased computational complexity. We believe that future research of this dissertation will play important role to successfully develop upcoming HEVC standard codec.

References

[1] Wikipedia- Broadband Internet Access
url : *http* ://*en.wikipedia.org*/*wiki*/*BroadbandInternetaccess* : *DSL*28*ADSL*2*FSDSL*.

[2] “Digital Video Processing”, Prentice Hall, 1995.

[3] Y. Wang, J. Ostermann, Y. Q. Zhang “Video processing and communications”, Prentice Hall, New Jersey, 2001.

[4] MPEG website: *url* : *http* : //*www.mpeg.org*.

[5] CCITT. Video Codec for Audiovisual services at p × 64 kbit/s, CCITT Recommendation H.261, 1990.

[6] ISO/IEC. MPEG-1: Information Technology - Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Mbit/s - Video, Geneva, 1993.

[7] ISO/IEC JTC1 and ITU-T. MPEG-2/H.262: Generic Coding of Moving Pictures and Associated Audio Information - Part 2: Video, November 1994.

[8] ITU-T. Video Coding for Low Bit-rate Communication, ITU-T Recommendation H.263, March 1996.

[9] ISO/IEC JTC1/SC29/WG11. MPEG-4 Video Verification Model: Version 18.0, WG11Document N3908, Pisa, January 2001.

[10] ITU-T. Video Coding for Low Bit-rate Communication, ITU-T Recommendation H.263, version 2 (H.263+), January 1998.

[11] ITU-T Recommendation H.263 - Annex U, "Video Coding for low bit rate communication - Annex U: Enhanced reference picture selection mode," Nov. 2000.

[12] ITU-T Recommendation H.263 - Annex V, "Video Coding for low bit rate communication - Annex V: Data-partitioned slice mode," Nov. 2000.

[13] ITU-T Recommendation H.263 - Annex W, "Video Coding for low bit rate communication - Annex W: Additional supplemental enhancement information specification," Nov. 2000.

[14] ISO/IEC JTC1/SC29/WG11. MPEG-4 Video Verification Model: Version 18.0, WG11 Document N3908, Pisa, January 2001.

[15] F. Pereira, editor. Signal Processing: Image Communication, volume 15. European Association for Signal Processing (EURASIP), January 2000.

[16] I.E.G. Richardson. Video Codec Design: Developing Image and Video Compression Systems. John Wiley and Sons Ltd., 2002.

[17] ITU-T and ISO/IEC JTC1. Advanced Video Coding for Generic Audiovisual Services, ITU-T Recommendation H.264 ISO/IEC 14496-10 AVC, 2003.

[18] I.E.G. Richardson. H.264 and MPEG-4 Video Compression. John Wiley and Sons Ltd., 2003.

[19] T. Wiegand, G.J. Sullivan, G. Bjontegaard, and A. Luthra. Overview of the H.264/AVC Video Coding Standard. IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, no. 7, pp. 560–576, July 2003.

[20] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, and G.J. Sullivan. Rate Constrained Coder Control and Comparison of Video Coding Standards. IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, no.7, pp. 688–703, July 2003.

[21] G. Sullivan (Rapporteur), Progress report and agenda items for Question 15 – advanced video coding," in SG16/Q15, Document Q15status. Geneva, Switzerland: ITU-T, January 1998.

[22] "Emerging H.264 Standard: Overview and TMS320DM642-Based Solutions for real time video application," White paper
$url : http : //focus.ti.com/pdfs/vf/vidimg/ubvideoh.26lwhitepaperv24.pdf.$

[23] Chen, M.C. and Willson, A.N., " Rate-distortion optimal motion estimation algorithm for video coding", IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-96), pp. 2096 – 2099, vol. 4, May 1996.

[24] T. Wiegand (Editor) and G. Sullivan (Editor), Joint draft international standard (FDIS) of joint video specification (ITU-T rec. H.264 ISO/IEC 14496-10 AVC)," in JVT, 7th Meeting, Document JVT-G050. Pattaya, Thailand: ITU-T, ISO/IEC, March 2003.

[25] Soon-kak Kwon a, A. Tamhankar b, K.R. Rao c, "Overview of H.264/MPEG-4 part 10", Journal of Visual Communication and Image representation, vol. 17, pp. 186-216, 2006.

[26] G. Sullivan (Rapporteur), Meeting report of the tenth meeting (meeting J) of the ITU-T Q.15/16 advanced video coding experts group," in SG16/Q15, 9. Meeting, Document Q15-J78. Osaka, Japan: ITU-T, May 2000.

[27] G. Bjontegaard (Editor), H.26L test model long term 1," in SG16/Q15 VCEG, 8. Meeting, Document Q15-H36. Berlin, Germany: ITU-T, August 1999.

[28] T. Wiegand (Editor), Joint model number 1, revision 1," in JVT, 1st Meeting, Document JVT-A003. Pattaya, Thailand: ITU-T, ISO/IEC, December 2001.

[29] T. Wiegand (Editor), H.26L test model long term 9," in SG16/Q6 VCEG, 14. Meeting, Document VCEG-N83. Santa Barbara, CA: ITU-T, September 2001.

[30] T. Wiegand and G. Sullivan (Editor), Joint draft international standard (FDIS) of joint video specification (ITU-T rec. H.264 ISO/IEC 14496-10 AVC)," in JVT, 7th Meeting, Document JVT-G050. Pattaya, Thailand: ITU-T, ISO/IEC, March 2003.

[31] G. Sullivan, P. Topiwala, A. Luthra, "The H.264/AVC advanced video coding standard: overview and introduction to the fidelity range extensions", SPIE Conference on Applications of Digital Image Processing XXVII, 2004.

[32] M. Karczewicz and R. Kurçeren: "The SP and SI Frames Design for H.264/AVC," IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, no. 7, pp. 637-644, July 2003.

[33] Soon-kak Kwon a, A. Tamhankar b, K.R. Rao c, "Overview of H.264/MPEG-4 part 10", Journal of Visual Communication and Image representation, vol. 17, pp. 186-216, 2006.

[34] G. Sullivan, P. Topiwala, A. Luthra, "The H.264/AVC advanced video coding standard: overview and introduction to the fidelity range extensions", SPIE Conference on Applications of Digital Image Processing XXVII, 2004.

[35] H. G. Musmann, P. Pirsh, and H. J. Grallert, "Advances in picture coding," Proceedings of the IEEE, vol. 73, no. 4, pp. 523-548, April 1985.

[36] E. Chan, S. Panchanathan, "Motion estimation architecture for video compression", IEEE Transaction on Consumer Electronics, vol. 39, pp. 292-297, Aug. 1993.

[37] J. Jain and A. K. Jain, "Displacement measurement and its application in inter frame image coding," IEEE Transaction on Communications, vol.29, pp.1799- 1808, Dec. 1981.

[38] N. Ahmed, T. Natarajan, and K. Rao, "Discrete cosine transform," IEEE Transaction on Computers, vol. 23, no. 1, pp. 90-93, January 1974 .

[39] K. R. Rao and R. Yip, "Discrete cosine transform – algorithms, advantages, applications," New York: Academic Press, 1990.

[40] H.S. Malvar, A. Hallapuro, and M.Karzewicz, " Low-complexity transform and quantization in H.264/AVC," IEEE Transaction on Circuits and Systems for Video. Technology, vol. 13, pp 598-603, July 2003.

[41] G. J. Sullivan and T. Wiegand, “Rate-distortion optimization for video compression,” IEEE Signal Processing Magagine, vol. 15, pp. 74-90, November 1998.

[42] M. C. Chen and A. N. Willson, Jr. “Rate-distortion optimal motion estimation algorithms for motion-compensated transform video coding,” IEEE Transaction on Circuits and Systems for Video Technology, 147-158, vol. 8, April 1998.

[43] Jun Zhang ,Xiaoquan Yi, Nam Ling, Weijia Shang, “Bit rate distribution for motion estimation in H.264 coding”, IEEE Transaction On Consumer Electronics, vol. 52, no. 2, pp. 606-610, May 2006.

[44] Jiangtao Wen, Mou Xiao, Jianwen Chen and Pin Tao “Fast Rate-Distortion Optimized Quantization for H.264/AVC”, Proceeding of the IEEE international Conference on Data Compression, pp. 557, Snowbird, Utah, March 2010.

[45] Q. Chen, Y. He, “A fast bits estimation method for rate-distortion optimization in H.264/AVC”, Proceedings of Picture Coding System, December 2004.

[46] D. Marpe, H.Schwarz, and T Wiegand, “Context-based adaptive binary coding in the H.264/AVC video compression standard,” IEEE Transaction on Circuits and System for Video Technology, vol. 13, no 7, pp. 620-636, July 2003.

[47] H.264/MPEG-4 AVC, From Wikipedia, the free encyclopedia, available online: $url : http : //en.wikipedia.org/wiki/Deblockingfilter.$

[48] G. Sullivan, P. Topiwala, A. Luthra, “The H.264/AVC advanced video coding standard: overview and introduction to the fidelity range extensions”, SPIE Conference on Applications of Digital Image Processing XXVII, 2004.

[49] C.H. Tseng, H.M. Wang, and J. F. Yang, “Enhanced Intra 4x4 Mode Decision for H.264/AVC Coders,” IEEE Trans. Circuits and System for Video Technology, vol. 16, no. 8, pp. 1027-1032, August 2006.

[50] Yu-Ming Lee, Yu-Ting Sun, and Yinyi Lin “SATD-Based Intra Mode Decision for H.264/AVC Video Coding”, IEEE Trans. Circuits and Systems for Video Technology, vol. 20, no. 3, pp.463-469, March 2010.

[51] ITU-T Recommendation H.264 and ISO/IEC 14496-10, “Advanced video coding for generic audiovisual services,” May 2003.

[52] Meng, B., and Au, O. C., “Fast intra-prediction mode selection for 4×4 blocks in H.264”, Internaitonal Conference on Image Processing, Barcelona, Spain, pp.389-392, 2003.

[53] Kim, C. S., Shih, H.-H., and Kuo, C. C. J., “Feature-based intra-prediction mode decision for H.264”, Proc. International Conference on Image Processing, Singapore, Vol.2, pp.769-772, 2004.

[54] Min-hua Zhou, “Intra prediction with simplified prediction modes”, ISO/IEC JT1/SC29/WG11 and ITU-T SG16 Q.6, JVT 4th Meeting, JVT-D026, Klagenfurt, Austria, pp.22-26, 2002.

[55] Kim, W.,-S., Cho, D.-S., and Kim, K. –W., “Intra prediction enhancements”, ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6, JVT 5th Meeting, JVT-E051, Geneva, Switzerland, pp.9-17, 2002.

[56] Shengsheng Yu, Yi Gao, Jiazhong Chen, and Jingli zhou, Distance-based weighted prediction for H.264 Intra Coding, IEEE IET International Conference on Audio, Language and Image Processing, pp.1477-1480, 2008.

[57] Nan Zhang, Baocai Yin, Dehui Kong, and Wenying Yue, “Spatial prediction based intra-coding,” IEEE International Conference on Multimedia and Expo (ICME), Taipei, Taiwan, pp.97-100, June 2004.

[58] Baocai Yin, Lei Sun, Dehui Kong, and Pengfei Ji, “Adjacent piexels-based intra prediction algorithm,” Journal of Beijing University of Technology, Vol.32, No.11, pp.1037-1042, November 2006.

[59] JM reference software,
$url : http : //iphome.hhi.de/suehring/tml/$

[60] Final draft of information technology – advanced coding of audio and video – part 2: video, AVS workgroup Doc. N1214, Shanghai, China, September 2005.

[61] ISO/IEC JTC 1/SC 29/WG 11: Coding of audio-visual objects – Part 2: Visual Amendment 1(MPEG-4), N3056, Hawaii (Maui), US, 1999.

[62] Jia Jie, Jung Eun-Ku, Kim Hae-Kwang, “Adaptive Transform Coefficient Scan for H.264 Intra Coding, IEICE Transaction (E90-D),” No. 10, pp. 1709-1711, October 2007.

[63] Y.-L. Lee, K.-H. Han, D.-G. Sim, and J. Seo, “Adaptive Scanning for H.264/AVC Intra Coding,” ETRI Journal, vol. 28, no. 5, pp. 668-667, October 2006.

[64] Byeong-Doo Choi, Jin-Hyung Kim, and Sung-Jea Ko, "Adaptive Coefficient Scanning Based on the Intra Prediction Mode," ETRI Journal, vol.29, no.5, pp.694-696, October 2007.

[65] X.Fan, Y.Lu and W.Gao, “A Novel Coefficient Scanning Scheme For Directional Spatial Prediction-Based Image Compression,” IEEE International Conference on Multimedia & Expo (ICME), pp.II/557- 560, July 2003.

[66] Li Zhang, Wen Gao, Qiang Wang and Debin Zhao, "Macroblock-Level Adaptive Scan Scheme for Discrete Cosine Transform Coefficients," IEEE International Symposium on Circuits and Systems, vol., no., pp.537-540, 27-30 May 2007.

[67] F. Pan, X. Lin, R. Susano, K. P. Lim, Z. G. Li, G. N. Feng, D. J. Wu, and S. Wu, “Fast mode decision for intra prediction,” Joint Video Team (JVT), Doc. JVT-G013, March 2003.

[68] K. P. Lim, S .Wu, D. J. Wu, S. Rahardja, X. Lin, F. Pan, and Z. G. Li, “Fast inter mode selection”, Joint Video Team (JVT), Doc. JVT-I20, September 2003.

[69] B. Jeon and J. Lee, “Fast mode decision for H.264,” Joint Video Team (JVT), Documents JVT-J033, December 2003.

[70] Y. F. Liang, I. Ahmad, J. Luo, Yu Sun, V. Swaminathan, “On using hierarchical motion history for motion estimation in H.264/AVC”, IEEE Transaction on Circuits and System for video technology, pp 1594-1603, Volume 15, Issue 12, December 2005.

[71] Ming Yang, Wensheng Wang, “Fast Macroblock mode selection based on motion content classification in H.264/AVC”, IEEE International Conference on Image Processing (ICIP), pp. 741-744, October 2004.

[72] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, and G. J. Sullivan, “Rate-constrained coder control and comparison of video coding standards”, IEEE Transaction on Circuits and System for Video Technology, vol. 13, no.7, pp. 688-703, July 2003.

[73] A. Hallapuro, M. Karczewicz, “Low Complexity Transform and Quantization –Part I: Basic Implementation”, ITU-T Q.6/SG16 Document JVT-B038, January 2002.

[74] J. S. Park and H. J. Song, “Fast selective intra mode decision in H.264/AVC”, IEEE Proceeding on Distributed Computing, Computer Communications (Networks),Parallel Computing, pp 1068-1072, 2005.

[75] F. Pan, X. Lin, S. Rahardja, K. P. Lim, Z. G. Li, D. Wu, S. Wu, “Fast intra mode decision algorithm for H.264/AVC video coding”, IEEE Proceedings on International Conference on Image Processing, pp781-784, 2004.

[76] C. P. Fan and J. F. Yang, “Fast center weighted hadamard transform algorithms”, IEEE Trans. on Circuit and Systems, PartII, Analog and Digital Signal Processing, vol.45, no.3, pp. 429-432, Mar 1998.

[77] C. P. Fan and J. F. Yang, “Fixed-pipeline 2D Hadamard Transform algorithms”, IEEE Transaction on Signal Processing, vol.45, no. 6, pp. 1669-1674, June 1997.

[78] I. E. G. Richardson, “H.264 and MPEG-4 video compression: video coding for next-generation multimedia”, Chichester: John Wiley and Sons, c 2003.

[79] J. Watkinson, “MPEG-2”, Oxford, Boston: Focal Press, 1999.

[80] Pereira, C. N. Fernando, “The MPEG-4 book”, Upper Saddle River, N.J.: Prentice Hall PTR, c2002.

[81] V. Roden, “H.261 and MPEG1-a comparison”, in Proc. 1996 IEEE Internationa Conference on Computers and Communications, pp. 65-71, March 1996.

[82] K. Y. Liao, J. F. Yang, and M. T. Sun, Rate-Distortion Cost Estimation for H.264/AVC¨, IEEE Transaction on Circuits and Systems for Video Technology, vol. 20, no.1, pp. 38-49, January 2010.

[83] Y. M. Lee, Y. T. Sun, and Y. Lin, SATD-Based Intra Mode Decision for H.264/AVC Video Coding¨, IEEE Transaction on Circuits and Systems for Video Technology, vol. 20, no. 3, pp. 463-469, March 2010.

[84] X. Jing and L. P. Chau, “Fast approach for H.264 inter mode decision,” Electronins Letters, vol. 40, no. 17, pp. 1051-1052, August 2004.

[85] D. Wu, F. Pan, K. P. Lim and S. Wu et al., “Fast intermode decision in H.264/AVC video coding,” IEEE Transaction on Circuits and Systems for Video Technology, vol. 15, no. 7, pp.953-958, July 2005.

[86] Y. H. Kim, J. W. Yoo, S. W. Lee et al., “Adaptive mode decision for H.264 encoder”, Electronics Letters, vol. 40, no. 19, pp. 1172 - 1173, September 2004.

[87] T. Y. Kuo and C. H. Chan, “Fast variable block size motion estimation for H.264 using likelihood and correlation of motion field”, IEEE Transaction on Circuits and Systems for Video Technology, vol. 16, no. 10, pp.1185-1195, Oct. 2006.

[88] C. Grecos, M. Yang, “Fast Mode Prediction for the Baseline and Main Profiles in the H.264 Video Coding Standard”, IEEE Transaction on Multimedia, vol. 8, pp. 1125-1134, December 2006.

[89] J. L. Fan, Y. Chen, X. D. Zhang, “A Novel Fast Inter Mode Decision for H.264”, Proceeding of IEEE Signal Processing 2006, vol. 2, 2006.

[90] S. Gao, T. J. Lu, “An improved fast mode decision algorithm in H.264 for video communications”, IEEE International Symposium on Symbolic and Algebraic Computation (ISSAC), January 2006.

[91] Z. P. Lin, H. T. Yu, F. Pan, “A scalable fast mode decision algorithm for H.264”, Proc. IEEE International Symposium on Circuits and System (ISCAS), May 2006.

[92] Z. Y. Wei, K. N. Ngan, “A Fast Macroblock Mode Decision Algorithm for H.264”, Proc. IEEE Asia Pasific Conference on Circuits and Syatems (APCCAS), pp. 772-775, December 2006.

[93] A. Hallapuro, M. Karczewicz, “Low Complexity Transform and Quantization –Part I: Basic Implementation”, ITU-T Q.6/SG16 Document JVT-B038, January 2002.

[94] I. Choi, J. Lee, B. Jeon, “Fast coding mode selection with rate-distortion optimization for MPEG-4 Part-10 AVC/H.264”, IEEE Transaction on Circuits and Systems for Video Technology, vol. 16, pp. 1557-1561, July 2006.

[95] X. Jing and L. P. Chau. “Fast approach for H.264 inter mode decision.” Electronics Letters, vol. 40, no. 17, pp. 1051-1052, August 2004.

[96] T. Y. Kuo and C. H. Chan, “Fast variable block size motion estimation for H.264 using likelihood and correlation of motion field”. IEEE Transaction on Circuits and Systems for Video Technology, vol. 16, no. 10, pp. 1185-1195, October 2006.

[97] Z. Y. Wei, K. N. Ngan, “A Fast Macroblock Mode Decision Algorithm for H.264”. Proceeding of IEEE Signal Processing 2005, vol. 1, pp. 243-246, August 2005.

[98] Y. M. Lee, Y. T. Sun, and Y. Lin, “SATD- Based Intra Mode Decision for H.264/AVC Video Encoding”, IEEE, Trans. on Circuits and Systems for Video Technology, vol 20, no. 3, pp. 463-469, March 2010.

[99] Md. Salah Uddin Yusuf and Mohiuddin Ahmad, “Relative SAD- Based Inter Mode Decision Algorithm for H.264/ AVC VideoStandard,” Journal of Multimedia, vol. 6, No. 5, pp. 425-435, October 2011.

[100] Md. Salah Uddin Yusuf and Mohiuddin Ahmad, "Modified Intra 4×4 Prediction Mode Scheme For H.264/AVC Video Coding", Proceedings of the International Conference on Engineering Research, Innovation and Education (ICERIE), pp. 648-651, 2011.

[101] Md. Salah Uddin Yusuf and Mohiuddin Ahmad, “A New Z-Scanning Scheme for Directional Spatial Prediction of AVS Intra Coding.” Proceedings of the International Conference on Computer and Information Technology (ICCIT), 2010.

[102] Md. Salah Uddin Yusuf and Mohiuddin Ahmad, “A Novel Scanning Scheme for Directional Spatial Prediction of AVS Intra coding”, International Journal of Advanced Research in Electrical, Electronic and Instrumentation Engineering (IJAREEIE), vol.2, no. 8, pp. 4120-4126, August 2013.

[103] Md. Salah Uddin Yusuf and Mohiuddin Ahmad, “A Novel Rate-Based Macroblock Classification for Fast Mode Decision in H.264 Video Coding”, Proceedings of the International Conference on Computer and Information Technology (ICCIT), 2011.

[104] Md. Salah Uddin Yusuf and Mohiuddin Ahmad, “A Novel Rate based Macroblock Classification for Fast Mode Decision in h.264 Video Coding”, International Journal of Advanced Research in Electrical, Electronic and Instrumentation Engineering (IJA-REEIE), vol.2, no. 8, pp. 4111-4119, August 2013.

[105] Md. Salah Uddin Yusuf and Mohiuddin Ahmad, “Relative SAD-based Inter Mode Decision Algorithm for H.264/AVC Video Standard,” Proceedings of the International Conference on Computer and Information Technology (ICCIT), 2010.

[106] Md. Salah Uddin Yusuf and Mohiuddin Ahmad, "Fast Inter Mode Decision Algorithm For H.264/AVC Video Standard, Proceedings of the International Conference on Engineering Research, Innovation and Education (ICERIE),pp. 590-595, 2011.

[107] Md. Salah Uddin Yusuf, M. I. Hossain and Mohiuddin Ahmad, “Probability Based Fast Inter Mode Decision Algorithm for H.264/AVC Video Standard”, International Forum on Strategic Tchnology (IFOST), 2014.

[108] Md. Salah Uddin Yusuf, M. I. Hossain and Mohiuddin Ahmad, “A New Fast Inter Mode Decision Algorithm in H.264/AVC Video Coding”, 8th International Conference on Electrical and Computer Engineering (ICECE), pp. 278-281 December, 2014.

[109] Md. Salah Uddin Yusuf, M. I. Hossain and Mohiuddin Ahmad, M. M. Rahman,N. Ahmed and M. A. Rashid, “Probability Based Fast Inter Mode Decision Algorithm for H.264/AVC Video Standard”, in International Journal of Applied Mathematics & Information Sciences, vol. 10, no.3, pp. 1071-1079, 2016.

[110] Md. Salah Uddin Yusuf and Mohiuddin Ahmad, “ Efficient Search Point Reduction and Ranking of commonly used Motion Estimation Algorithms for Video coding”, Proceedings of the 2nd International Conference on Electrical Engineering and Information Communication Technology (ICEEICT 2015).

[111] Md. Salah Uddin Yusuf, M. Islam, Mohiuddin Ahmad, "Fast and Efficient Inter Mode Decision Algorithm for H.264/AVC Encoding Optimization," in International Journal of Electronics Communication and Computer Engineering, vol. 6, no. 6, pp. 746-751, 2015.

[112] Md. Salah Uddin Yusuf, M. Islam, Mohiuddin Ahmad, "Efficient Search Points Reduction and Ranking of Motion Estimation Algorithms in Video Coding," in International Journal of Electronics Communication and Computer Engineering, vol. 6, no. 6, pp. 739-745, 2015.

[113] Chen- Kuo Chiang and Shag-Hong Lai, “Fast H.264 Encoding Based on Statistical Learning,” PCM 2010, Part II, LNCS 6298, pp. 179-189, 2010.

[114] Ma, W. , Yang, S. Gao L., Pei, C., Yan, S. , “ Fast Mode Selection Scheme for H.264/ AVC Inter Prediction Based on Statistical Learning Method,” IEEE International Conference on Multimedia and Expo (ICME), 2009.

[115] Turaga, D.S.,Tsuhan Chen, “Estimation and Mode Decision for Spatially Correlated Motion Sequences” IEEE Transactions on Circuits and Systems for Video Technology, Vol. 11 , pp. 1098-1107 , October 2001.

[116] Y. H. Kim, J. W. Yoo, S. W. Lee et al., “Adaptive mode Decision for H.264 encoder”, Electronics Letters, Vol. 40, no. 19, pp. 1172-1173, September 2004.

[117] T. Y. Kuo and H.J Lu, “Efficient h.264 encoding based on skip mode early termination” Proceedings of the First Pacific Rim conference on Advances in Image and Video Technology, pp. 761-770, Springer-Verlag Berlin, Heidelberg , 2006.

[118] R. Srinivasan and K. R. Rao, “Predictive coding based on efficient motion estimation,” IEEE Transaction on Communications, vol. 33, no. 8, pp. 888–896, August 1985.

[119] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, “Motion compensated interframe coding for video conferencing,” in Proceeding of National Telecommunication Conference, New Orleans, LA, pp. G5.3.1-G5.3.5, December 1981.

[120] I. Ahmad, W. Zheng, J. Luo, and M. Liou, “A fast adaptive motion estimation algorithm,” IEEE Transaction on Circuits and Systems Video Technology, vol. 16, no. 3, pp. 420–438, March 2006.

[121] J. Kim and T. Park. A novel VLSI architecture for full Search variable block-Size motion estimation. IEEE Transactions on Consumer Electronics, vol.55, no. 2, pp. 728-733, 2009.

[122] J.-C. Tuan, T.-S. Chang, and C.-W.Jen, “On the data reuse and memory bandwidth analysis for full-search block-matching VLSI architecture,” IEEE Transaction on Circuits and Systems for Video Technology, vol. 12, pp. 61–72, January 2002.

[123] S.-S. Lin, P.-C. Tseng, and L.-G. Chen, “Low- power parallel tree architecture for full search block-matching motion estimation,” in Proc. IEEE International Symposium on Circuits and Systems, pp. 313–316, May 2004.

[124] S.B. Lopes, I. S. Silva and L.V Agostini, “An efficient memory hierarchy for full search motion estimation on high definition digital videos,” Proceedings of the 24th symposium on Integrated circuits and systems design, pp.131-136, August 2011.

[125] M. Mohammadzadeh, M. Eshghi, M. M. Azadfar, “Parameterizable implementation of full search block matching algorithm using FPGA for real-time applications,” 5th International Caracas Conference on Devices Circuits and Systems(ICCDCS), pp. 200–203, 2004.

[126] R. Li, B. Zeng, and M. L. Lio, “A new three-step search algorithm for block motion estimation,” IEEE Trans. Circuits and Systems for Video Technology, vol. 4, no. 4, pp. 438–443, August 1994.

[127] L. M. Po and W. C. Ma, “A novel four-step search algorithm for fast block motion estimation,” IEEE Trans. Circuits and Systems for Video Technology, vol. 6, no. 3, pp. 313–317, June 1996.

[128] L. K. Liu and E. Feig, “A block-based gradient descent search algorithm for block motion estimation in video coding,” IEEE Transaction on Circuits and Systems for Video Technology, vol. 6, no. 4, pp. 419–422, August 1996.

[129] C. H. Cheung and L. M. Po, “A novel cross-diamond search algorithm for fast block motion estimation,” IEEE Transaction on Circuits and Systems for Video Technology, vol. 12, no. 12, pp. 1168–177, December 2002.

[130] C. H. Cheung and L. M. Po, “Novel cross-diamond-hexagonal search algorithms for fast block motion estimation,” IEEE Transaction on Multimedia, vol. 7, no. 1, pp. 16–22, February 2005.

[131] J. Y. Tham, S. Ranganath, M. Ranganath, and A. A. Kassim, “A novel unrestricted center-biased diamond search algorithm for block motion estimation,” IEEE Transaction on Circuits and Systems for Video Technology, vol. 8, no. 4, pp. 369–377, August 1998.

Appendix A

Encoder Configuration

In H.264/AVC reference software encoder parameters are adjusted by *encoder.cfg* file. The follow shows part of the *encoder.cfg* file of JM17.0 reference software.

```
# New Input File Format is as follows
# <ParameterName> = <ParameterValue> # Comment
# See configfile.h for a list of supported ParameterNames
##################################################################################
#Files
##################################################################################
InputFile            = "Stefan_ cif.yuv"   # Input sequence
InputHeaderLength    = 0                   # If the inputfile has a header, state it's length
                                             in byte here
StartFrame           = 0                   # Start frame for encoding. (0-N)
FramesToBeEncoded    = 50                  # Number of frames to be coded
FrameRate            = 30.0                # Frame Rate per second (0.1-100.0)
SourceWidth          = 176                 # Frame width
SourceHeight         = 144                 # Frame height
TraceFile            = "trace_enc.txt"
ReconFile            = "test_rec.yuv"
OutputFile           = "test.264"
```

```
##############################################################################
#Encoder Control
##############################################################################
ProfileIDC             = 66   # Profile IDC (66=baseline, 77 =main, 88=extended);
                              FREXT Profiles: 100=High, 110=High 10, 122=High 4:2:2,
                              144=High 4:4:4, for parameters see below)
LevelIDC               = 30   # Level IDC   (e.g. 20 = level 2.0)
IntraPeriod            = 4    # Period of I-Frames (0=only first)
IDRIntraEnable         = 0    # Force IDR Intra  (0=disable 1=enable)
QPISlice               = 28   # Quant. param for I Slices (0-51)
QPPSlice               = 28   # Quant. param for P Slices (0-51)
FrameSkip              = 0    # Number of frames to be skipped in input (e.g 2 will code
                              every third frame)
ChromaQPOffset         = 0    # Chroma QP offset (-51……..51)
UseHadamard            = 0    # Hadamard transform (0=not used, 1=used)
SearchRange            = 32   # Max search range
NumberReferenceFrames  = 1    # Number of previous frames used for inter motion search
                              (1-16)
PList0References       = 0    # P slice List 0 reference override (0 disable, N <=
                              Numberof ReferenceFrames)
Log2MaxFrameNum        = 4    # Sets log2_max_frame_num_minus4 (0-3 : based on
                              FramesToBeEncoded, >3 : Log2MaxFrameNum - 4)
MbLineIntraUpdate      = 0    # Error robustness(extra intra macro block
                              updates)(0=off,N: One GOB every N frames are intra
                              coded)
RandomIntraMBRefresh   = 0    # Forced intra MBs per picture
InterSearch16x16       = 1    # Inter block search 16x16 (0=disable, 1=enable)
InterSearch16x8        = 1    # Inter block search 16x8  (0=disable, 1=enable)
InterSearch8x16        = 1    # Inter block search  8x16 (0=disable, 1=enable)
InterSearch8x8         = 1    # Inter block search  8x8  (0=disable, 1=enable)
InterSearch8x4         = 1    # Inter block search  8x4  (0=disable, 1=enable)
InterSearch4x8         = 1    # Inter block search  4x8  (0=disable, 1=enable)
InterSearch4x4         = 1    # Inter block search  4x4  (0=disable, 1=enable)
UseFME                 = 1    # Use fast motion estimation (0=disable, 1=enable)
```

```
################################################################################
#B Slices
################################################################################
NumberBFrames          = 0                  # Number of B coded frames inserted (0=not used)
QPBSlice               = 30                 # Quant. param for B slices (0-51)
BRefPicQPOffset        = 0                  # Quantization offset for reference B coded pictures
                                            (-51..51)
DirectModeType         = 1                  # Direct Mode Type (0:Temporal 1:Spatial)
DirectInferenceFlag    = 1                  # Direct Inference Flag (0: Disable 1: Enable)
BList0References       = 0                  # B slice List 0 reference override (0 disable, N <=
                                            NumberReferenceFrames)
                                            # B slice List 1 reference override (0 disable, N <=
                                            NumberReferenceFrames)
                                            # 1 List1 reference is usually recommended for
BList1References       = 1                  normal GOP Structures.
                                            # A larger value is usually more appropriate if a
                                            more flexible
                                            # structure is used (i.e. using PyramidCoding)
BReferencePictures     = 0                  # Referenced B coded pictures (0=off, 1=on)
PyramidCoding          = 0                  # B pyramid (0= off, 1= 2 layers, 2= 2 full pyramid,
                                            3 = explicit)
                                            # Explicit Enhancement GOP. Format is
                       = "b1r30b3r20b2      {FrameDisplay_orderReferenceQP}.
ExplicitPyramidFormat
                       e34b0e30b4r22"       # Valid values for reference type is r:reference,
                                            e:non reference.
PyramidRefReorder      = 1                  # Reorder References according to Poc distance for
                                            PyramidCoding (0=off, 1=enable)
PocMemoryManagement    = 1                  # Memory management based on Poc Distances for
                                            PyramidCoding (0=off, 1=on)
```

```
##############################################################################
#SP Frames
##############################################################################
SPPicturePeriodicity   = 0      # SP-Picture Periodicity (0=not used)
QPSPSlice              = 28     # Quant. param of SP-Slices for Prediction Error (0-51)
QPSP2Slice             = 27     # Quant. param of SP-Slices for Predicted Blocks (0-51)

##############################################################################
Output Control, NALs
##############################################################################
SymbolMode             = 1      # Symbol mode (Entropy coding method: 0=UVLC,
                                1=CABAC)
OutFileMode            = 0      # Output file mode, 0:Annex B, 1:RTP
PartitionMode          = 0      # Partition Mode, 0: no DP, 1: 3 Partitions per Slice

##############################################################################
#CABAC context initialization
##############################################################################
ContextInitMethod      = 0      # Context init (0: fixed, 1: adaptive)
FixedModelNumber       = 0      # model number for fixed decision for inter slices ( 0, 1, or
                                2 )

##############################################################################
#Interlace Handling
##############################################################################
PicInterlace           = 0      # Picture AFF    (0: frame coding, 1: field coding,
                                2:adaptive frame/field coding)
                                #Macroblock AFF (0: frame coding, 1: field coding,
MbInterlace            = 0      2:adaptive frame/field coding, 3:combined with
                                PicInterlace=0, to do frame MBAFF))
IntraBottom            = 0      # Force Intra Bottom at GOP Period

##############################################################################
#Weighted Prediction
##############################################################################
WeightedPrediction     = 0      # P picture Weighted Prediction (0=off, 1=explicit mode)
WeightedBiprediction   = 0      # B picture Weighted Prediciton (0=off, 1=explicit mode,
                                2=implicit mode)
UseWeightedReference
ME                     = 0      # Use weighted reference for ME (0=off, 1=on)
```

```
##########################################################################################
#Loop filter parameters
##########################################################################################
LoopFilterParametersFlag = 0   # Configure loop filter (0=parameter below ingored,
                               1=parameters sent)
LoopFilterDisable        = 0   # Disable loop filter in slice header (0=Filter, 1=No Filter)
LoopFilterAlphaC0Offset  = 0   # Alpha & C0 offset div. 2, {-6, -5, ... 0, +1, .. +6}
LoopFilterBetaOffset     = 0   # Beta offset div. 2, {-6, -5, ... 0, +1, .. +6}

##########################################################################################
#Error Resilience / Slices
##########################################################################################
SliceMode                  = 0     # Slice mode (0=off 1=fixed #mb in slice 2=fixed
                                   #bytes in slice 3=use callback)
SliceArgument              = 50    # Slice argument (Arguments to modes 1 and 2
                                   above)
num_slice_groups_minus1    = 0     # Number of Slice Groups Minus 1, 0 == no FMO,
                                   1 == two slice groups, etc.
                                   # 0:  Interleave, 1: Dispersed,    2: Foreground
                                   with left-over,
slice_group_map_type       = 0     # 3:  Box-out,    4: Raster Scan   5: Wipe
                                   # 6:  Explicit, slice_group_id read from
                                   SliceGroupConfigFileName
                                   # 0: box-out clockwise, raster scan or wipe right,
slice_group_change_direction
_flag                      = 0     # 1: box-out counter clockwise, reverse raster scan
                                   or wipe left
slice_group_change_rate_min
us1                        = 85    #
                           =
SliceGroupConfigFileName   "sg0conf.cfg" # Used for slice_group_map_type 0, 2, 6
UseRedundantSlice          = 0     # 0: not used, 1: one redundant slice used for each
                                   slice (other modes not supported yet)
```

```
################################################################################
#Search Range Restriction / RD Optimization
################################################################################
RestrictSearchRange   = 2    # restriction for (0: blocks and ref, 1: ref, 2: no restrictions)
                             # rd-optimized mode decision
                             # 0: RD-off (Low complexity mode)
RDOptimization        = 1    # 1: RD-on (High complexity mode)
                             # 2: RD-on (Fast high complexity mode - not work in FREX
                             Profiles)
                             # 3: with losses
DisableThresholding   = 0    # Disable Thresholding of Transform Coefficients (0:off, 1:on)
DisableBSkipRDO       = 0    # Disable B Skip Mode consideration from RDO Mode decision
                             (0:off, 1:on)
LossRateA             = 10   # expected packet loss rate of the channel for the first partition,
                             only valid if RDOptimization= 2
LossRateB             = 0    # expected packet loss rate of the channel for the second
                             partition, only valid if RDOptimization = 2
LossRateC             = 0    # expected packet loss rate of the channel for the third partition,
                             only valid if RDOptimization = 2
NumberOfDecoders      = 30   # Numbers of decoders used to simulate the channel, only valid if
                             RDOptimization = 2
RestrictRefFrames     = 0    # Doesnt allow reference to areas that have been intra updated in
                             a later frame.
```

```
#####################################################################################
#Additional Stuff
#####################################################################################
                                                        # If 1, Inter pixels are not used for
UseConstrainedIntraPred   = 0                           Intra macroblock prediction.
                                                        # Last frame number that have to be
LastFrameNumber           = 0                           coded (0: no effect)
                                                        # QP (I-slices)  for second part of
ChangeQPI                 = 16                          sequence (0-51)
                                                        # QP (P-slices)  for second part of
ChangeQPP                 = 16                          sequence (0-51)
                                                        # QP (B-slices)  for second part of
ChangeQPB                 = 18                          sequence (0-51)
                                                        # QP offset (stored B-slices)  for
ChangeQPBSRefOffset       = 2                           second part of sequence (-51..51)
                                                        # Frame no. for second part of
ChangeQPStart             = 0                           sequence (0: no second part)
NumberofLeakyBuckets      = 8                           # Number of Leaky Bucket values
                                                        # File from which encoder derives rate
LeakyBucketRateFile       = "leakybucketrate.cfg"       values
                                                        # File where encoder stores
LeakyBucketParamFile      = "leakybucketparam.cfg"      leakybucketparams
NumberFramesInEnhance                                   # number of frames in the Enhanced
mentLayerSubSequence      = 0                           Scalability Layer(0: no Enhanced
                                                        Layer)
NumberOfFrameInSecond                                   # Number of frames to be coded in the
IGOP                      = 0                           second IGOP
                                                        # (0: no spare picture info, 1: spare
SparePictureOption        = 0                           picture available)
                                                        # Threshold for spare reference
SparePictureDetectionThr  = 6                           pictures detection
                                                        # Threshold for the spare macroblock
SparePicturePercentageThr = 92                          percentage
                                                        # (0: POC mode 0, 1: POC mode 1, 2:
PicOrderCntType           = 0                           POC mode 2)
```

```
##############################################################################
#Rate control
##############################################################################
RateControlEnable    = 0       # 0 Disable, 1 Enable
Bitrate              = 45020   # Bitrate(bps)
                               # Initial Quantization Parameter for the first I frame
InitialQP            = 24      # InitialQp depends on two values: Bits Per Picture,
                               # and the GOP length
                               # Number of MBs in the basic unit
BasicUnit            = 11      # should be a fractor of the total number
                               # of MBs in a frame
ChannelType          = 0       # type of channel( 1=time varying channel; 0=Constant
                               channel)
```

Appendix B

Input Sequence

Input Sequence used for Research

Sequence Name & No. of frame	Sample frame of YUV Video Sequence	Sequence format 4:2:0 YUV
Akiyo 100 Frames		QCIF Format (176×144)
Foreman 150 Frames		QCIF Format (176×144)
Stefan 90 Frames		QCIF Format (176×144)
Mobile 300 Frames		QCIF Format (176×144)

Input Sequence used for Research

Sequence Name & No. of frame	Sample frame of YUV Video Sequence	Sequence format 4:2:0 YUV
News 300 Frames		QCIF Format (176×144)
Bus Claire 494 Frames		QCIF Format (176×144)
Container 300 Frames		QCIF Format (176×144)
Salesman 449 Frames		QCIF Format (176×144)

Input Sequence used for Research

Sequence Name & No. of frames	**Sample frame of YUV Video Sequence**	**Sequence format 4:2:0 YUV**
Akiyo 100 Frames		CIF Format (352×288)
Foreman 300 Frames		CIF Format (352×288)
Stefan 90 Frames		CIF Format (352×288)
Bus 150 Frames		CIF Format (352×288)

Input Sequence used for Research

Sequence Name & No. of frames	Sample frame of YUV Video Sequence	Sequence format 4:2:0 YUV
Paris 1065 frames		CIF Format (352×288)
Football 125 Frames		SIF Format (352×240)
City 600 frame		4CIF Format (704×576)
Soccer 600 frame		4CIF Format (704×576)